Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

DIPLOMSKI RAD

Tomislav Slavina

Zagreb, 2012.

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

DIPLOMSKI RAD

Voditelj rada:

prof. dr. sc. Bojan Jerbić

Tomislav Slavina

Zagreb, 2012.

SADRŽAJ

SAŽETAK

U uvodnom dijelu ovog rada objasniti će se neki od osnovnih pojmova potrebnih za razumijevanje gradiva. Na početku biti će nekoliko riječi o ulogama robota u medicini općenito, od prvih, osnovnih, početaka, pa sve do današnjih složenih sustava. Budući da se cijeli rad programa zasniva na manipularanju DICOM formatom snimaka (zapis medicinskih podataka) , biti će nekoliko riječi i o tome. Biti će riječi i o programskom paketu Matlab i njegovim mogućnostima u kojem je i razvijena programska podrška „Brainiac“. Objasniti će se uporaba grafičkog korisničkog sučelja, jednog od paketa Matlaba, u kojem je sastavljen program tako da bi bio prilagođen korsnicima sa manje iskustva i znanja u programiranju.

Drugi dio rada bavi se problemima koji se javljaju prilikom procesiranja DICOM snimkama, neželjene nuspojave te različiti oblici šumova. Također, biti će objašnjeno kako su se ti problemi postepeno riješavali.

U trećem dijelu rada detaljno će se objasniti postupak izrade funkcija programa i njegovih korisnih alata uz slikovite primjere i isječke programskog koda te uz detaljan opis i objašnjenje načina rada programskog koda.

POPIS SLIKA

POPIS OZNAKA

MR	Magnetska rezonanca
DICOM	Digitalni prikaz slika i komunikacija u medicini
TCP/IP	Grupa protokola koji služe za mrežnu komunikaciju
MATLAB	Programski jezik visoke razine
DARPA	Agencija Ministarstva obrane SAD-a odgovorna za razvoj novih tehnologija za vojsku SAD-a.
NASA	Američka svemirska agencija
JPEG	Komprimirani slikovni format
ACR	Američki fakultet radiologije
NEMA	Nacionalna udruga električnih proizvođača

Izjava

Izjavljujem da sam ovaj rad radio samostalno koristeći se stečenim znanjem i navedenom literaturom

Zahvale:

Zahvaljujem prof. dr. sc. Bojanu Jerbiću na strpljenju, podršci i korisnim savjetima te povjerenju da ću ovaj rad završiti u roku.

Zahvaljujem mentoru, dr. sc. Petru Ćurkoviću na stručnim savjetima i nesebičnoj pomoći pri prikupljanju materijala korisnih za izradu ovog rada.

Zahvaljujem dr. sc. Tomislavu Stipančiću na pomoći oko filtriranja slika u svrhu otklanjanja pojave šuma

Zahvaljujem se obitelji koja je bila uz mene tijekom svih godina mog školovanja

Tomislav Slavina

1. Uvod

Čovjek je tijekom cijele svoje evolucije težio duljem životu i boljoj kvaliteti življenja. Danas je medicina puno napredovala za razliku od prije samo nekoliko stotina godina. Koriste se visokoučinkoviti lijekovi i pomagala koja značajno produljuju čovjekov životni vijek. Pitanje koje se nameće je može li medicina napredovati. Odgovor je: naravno da može. To se može ostvariti primjenom vrlo preciznih i pokretljivih robota. Uloga robota u suvrmenoj medicini danas je značajna. Roboti i robotska pomagala sve više nalaze svrhu pri zahtjevnim operacijama gdje se traži vrhunska preciznost, brzina, ponovljivost, pokretljivost te možda najbitnije povratna informacija s područja zahvata koja omogućava kirurzima procijeniti stanje tkiva te reagirati u skladu s time. Oni su u stanju operaciju obaviti brzo, sa manje rezova, manje opasnosti od infekcija i manju traumu za pacijenta. Širom svijeta operacije se odvijaju na način da kirurg smisli operativni zahvat te u samom zahvatu u pomoć priskače robot. U medicini mnoga područja zahtjevaju mirnu ruku i visoku preciznost. Jedno od takvih područja je neurokirurgija. Mozak je najsloženiji ljudski organ, te su upravo u neurokirurgiji zahtjvi za preciznost veći i do deset puta u odnosu na druge grane medicine. Upravo bi u takvim granama medicine roboti pronašli svoju stvarnu svrhu i vrijednost.

U neurokirurgiji prvi korak ka bilo kakvoj operaciji je skeniranje ljudske glave u svrhu dobivanja uvida u strukturu mozga. Na osnovu tih snimaka planira se operacija. Magnetska ili magnetna rezonancija (MR) je naziv uređaja u medicini kojim se prikazuju slojevi ljudskog tijela. Snimke su, zapravo, najčešće niz horizontalnih slojeva ljudske gleve, koji, naslagani jedan na drugi tvore cijelinu.

Postoji više formata snimaka, a u ovom radu koristit će se originalne snimke ljudske glave snimljene u Kliničkoj bolnici Dubrava u Zagrebu, formata DICOM (engl. Digital Imaging and Communications in Medicine). DICOM format je standardan format za spremanje, obradu, ispis i prijenos informacija u procesiranju medicinskih slika. Format uz informacije o slici (ime ustanove, ime pacijenta, datum, vrijeme, rezoluciju, razmak između piksela i sl.) sadrži i protokol za mrežnu komunikaciju. To znači da dva uređaja mogu mrežno izmjenjivati podatke o pacijentu pomoću TCP/IP protokola. [1]

Usporedno uz opremu razvija se i programska podrška koja je ključan faktor uspješne robotske operacije. Programska podrška je umjetna inteligencija robota da izvrši određeni zadatak. Prije bilo kakvog zahvata kirurg mora u obliku zadataka odrediti operativni zahvat robota asistenta. Da bi se to moglo uz robot mora doći i odgovarajuća računalna podrška u obliku kakvog programa ili slično kako bi bilo kakva kretnja robota bila moguća. To su kodirane informacije koje gvore mehaničkoj spravi (aktuatorima) koje zadtke vršiti te koje kontroliraju robotske postupke.

Programiranje robota nipošto nije trivijalno. Razlikujemo i programsku podršku za planiranje zahvata. Takav program pomaže operatoru planiranje zahvata i robotskih pokreta, te nema doticaja sa samim robotskim uređajem. Naravno, te dvije programske skupine mogu se međusobno uklopiti u jedan softver koji bi bio i za obradu podataka i planiranje, ta za samo izvršavanje zadatka putem robota. Program za potrebe ovog rada razvijao se u grafičkom korisničkom sučelju MATLAB programskog paketa.

2. Cilj zadatka

Cilj ovog zadatka je stvoriti programsku podršku za planiranje zahvata u ljudskom mozgu. Ideja je posložiti DICOM snimke jednu na drugu kreirajući tako volumen kojim možemo manipulirati. Iz tako dobivenog volumena možemo stvarati nove presjeke, ovoga puta u preostale dvije ravnine. Na taj način dobije se bolji uvid u pacijentovo stanje te se olakšava identifikacija problema te planiranje zahvata.

Programska podrška razvijala se u okviru programskog paketa MATLAB, snažnog matematičkog računalnog softvera za suvremene inženjere. Sadašnja programska podrška za obradu podataka za robotsku operaciju vrlo je skupa i teško je pronaći podršku koja bi točno odgovarala za zadavanje svih zadataka koji je robot sposoban učiniti. Stoga postoje razni programi primjenjivi u medicini za programiranje robota prikladni svaki za određeni zadatak.

Rezultat ovog rada je program „Brainiac“ koji omogućuje korisniku interaktivno istraživanje DICOM slojeva, uz mnoge alate koji pomažu vizualizaciji pacijentove glave i pristupu planiranja buduće operacije ili zahvata na pacijentu, kao što su 3D realni prikaz glave ili planiranje putanje uz prikaz prodora na odgovarajućim presjecima. Putanja se kreira uzimanjem dviju točaka sa nekog od presjeka ili sa površine glave realnog prikaza. Uz to program ima sposobnost ocijene veličine objekta na slici, kao i kalkulacije površine nekog područja na presjeku.

3. Roboti u medicini

3.1. Povijest robotske medicine

Laparoskopija je minimalno invazivna procedura kojom se u trbušnu šupljinu ulazi sa krutim cjevima kroz koje se može u trbušnu šupljinu uvesti teleskop sa malom kamerom i specijalni laparoskopski kirurški instrumenti. Prethodno se u trbušnu šupljinu upumpava, sa posebnim uređajem, ugljikov dioksid te se na taj način se formira virtualna šupljina u kojoj je moguće izvršiti operacije na više organa unutar trbušne šupljine. Kirurg proceduru ne vidi direktno nego na posebnom monitoru poput onoga na običnom kućnom računalu. Povijest laparoskopije seže i prije 200 godina, no prva suvremena operacija slijepog crijeva bila je 1980. godine. To su prve operacije koje se u svojim zahvatima ozbiljno oslanjaju na pomoćne uređaje. Usporedno sa komunikacijskim tehnologijama razvijaju se i komunikacijski uređaji za brz i učinkovit prijenos medicinskih podataka.

1985. godine robot PUMA 560 je bio korišten za postavljanje igle za biopsiju mozga koristeći CT navođenje. 1988. godine je PROBOT, razvijen na londonskom sveučilištu "Imperial College London", bio korišten za izvođenje kirurših zahvata prostate. ROBODOC tvrtke Integrated Surgical Systems uveden je 1992. godine za ugradnju preciznih elemenata u bedrenu kost kod zamjene kuka. Daljnji razvoj sustava rezultirao je kirurškim sustavom „da Vinci“.[3]

Slika 1. Robotski kirurški sustav „Da Vinci“[2]

Kirurški sustav da Vinci se sastoji od tri komponenete: kirurške konzole, operacijskih kolica pacijenta s 4 robotske poluge kojima upravlja kirurg (jedna za upravljanje kamerom i tri za upravljanje instrumentima), i 3D vizijskog sustava visoke rezolucije. Zglobni kirurški instrumenti su postavljeni na robotske poluge koje su uvedene u tijelo kroz kanale. Originalni robotski kirurški sustav na kojem je utemeljen da Vinci je razvijen u „SRI International“ u Menlo Park-u, California, s velikom podrškom od strane agencija DARPA i NASA. [3]

Iako je kirurški robot za kirurgiju na daljinu izvorno bio namijenjen za omogućavanje izvođenja kirurških zahvata na udaljenim ratištima i drugim opasnim okruženjima, na kraju se pokazalo da ima veću uporabnu vrijednost za minimalno invazivnu kirurgiju na licu mjesta. Patenti za prve prototipe su bili prodani tvrtki Intuitive Surgical iz Kalifornije.[3]

Da Vinci osjeća pokrete ruku kirurga i elektronički ih prevodi u skalirane mikro-pokrete specijalnih kirurških instrumenata. Da Vinci također prepoznaje i filtrira podrhtavanja pokreta ruku kirurga, tako da se ne prenose na vrhove instrumenata. Kamera koju koristi sustav daje stvarnu stereo sliku koja se prenosi na kiruršku konzolu. Kirurški sustav da Vinci je odobren od strane Američke agencije za hranu i lijekove (FDA) za različite kirurške zahvate uključujući operacije karcinoma prostate, histerektomije i poporavke mitralnog zaliska, i koristi se u više od 800 bolnica širom svijeta. Da Vinci sustav je korišten u 48,000 kirurških zahvata tijekom 2006. godine. Najnoviji model da Vinci HD Si na tržište je izbačen u travnju 2009. godine. Prvi kirurški zahvat izveden je na The Ohio State University Medical Center u Columbus, Ohio, pod vodstvom Dr. Robert E. Michler, profesora i pročelnika na kardio-torakalnoj kirurgiji.[3]

U rujnu 2010. godine izveden je prvi robotski zahvat na bedrenom krvožilnom sustavu na Sveučilišnom medicinskom centru Ljubljana (UMC Ljubljana), Slovenija. Istraživanje je vodioBorut Geršak, voditelj odjela za kadiovaskularnu kirurgiju u centru.

Geršak je objasnio da je korišteni robot bio prvi pravi robot u povijesti robotske kirurgije, što znači da korisničko sučelje nije poput kirurških instrumenata i robot nije samo imitirao pokrete ljudskih ruku, već je bio upravljan pritiscima na gumbe, slično kao u video igricama. Robot je bio uvezen u Sloveniju iz SAD-a. [3]

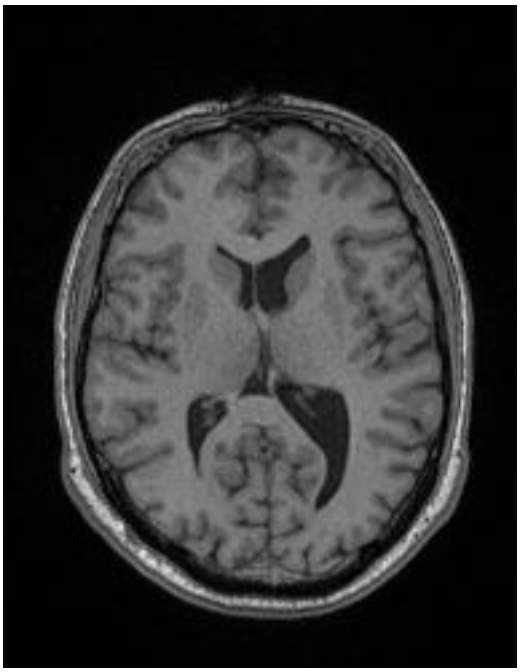
3.2. Roboti u kirurgiji

Robotski asistirana kirurgija je razvijena da prevlada organičenja minimalno invazivne kirurgije i da proširi mogućnosti kirurga u izvođenju otvorene kirurgije. U slučaju robotski asistirane minimalno invazivne kirurgije, umjesto direktnog pokretanja instrumenata, kirurg koristi neki od dva načina upravljanja instrumentima; ili direktno daljinsko rukovanje ili računalno upravljanje. Daljinski rukovatelj omogućuje kirurgu izvođenje kirurških pokreta, dok robotske ruke izvršavaju te pokrete koristeći izvršne elemente za izvođenje stvarnih kirurških akcija na pacijentu. Kod računalno upravljanih sustava kirurg koristi računalo za upravljanje robotskim rukama i izvršnim elementima, iako ti sustavi mogu koristiti i daljinske rukovatelje kao ulazne elemente.[3]

Glavni doprinosi kirurških robota su kirurgija na daljinu, minimalno invazivna kirurgija i automatska kirurgija bez čovjeka. To znači da kirurg ne mora biti prisutan, već se može nalaziti bilo gdje na svijetu a da i dalje dobije uvid u pacijentovo stanje i reagira na osnovu istog u trenu. Neke od najvažnijih prednosti robotske kirurgije su preciznost, minijaturizacija, manje incizije, manji gubitak krvi, manje boli, i brži oporavak. Daljnje prednosti su pokretljivost bolja od čovjekove ruke, trodimenzionalno uvećanje, koje rezultira poboljšanom ergonomijom. Robotska kirurgija pridonosi kraćem boravku pacijenata u bolnici, manjem gubitku krvi, manjem broju transfuzija, i manje korištenja lijekova za ublažavanje boli.[3]

4. Format slika DICOM

Cijeli rad prilagođen je radu sa snimkama DICOM formata (.dcm). DICOM format je standardan format za spremanje, obradu, ispis i prijenos informacija u procesiranju medicinskih slika. Uključuje definiciju formata i protokole za mrežnu komunikaciju. Protokol za mrežnu komunikaciju je aplikacijski protokol koji koristi TCP/IP (*engl*. Transmission Control Protocol/ Internet Protocol) za komunikaciju među sustavima. Dicom datoteke mogu se razmjenjivati između uređaja koji imaju sposobnost prihvaćanja i ispisa tog formata. Nacionalna udruga električnih proizviđača (*engl*. NEMA, National Electrical Manufacturers Association) posjeduje autorsko pravo na DICOM format. Razvijn je od strane Odbora za DICOM standarde čiji su članovi i dijelom članovi NEMA organizacije. DICOM omogućuje integraciju skenera, servera, printera, te mrežnog sučelja više različitih proizvođača u jedinstveni sustav arhiviranja i komunikacije snimaka. Različiti uređaji dolaze sa izjavama o stupnju korištenja DICOM snimaka te koje klase ti uređaji podržavaju. Format je široko prihvaćen od strane zdravstvenih ustanova te se sa istom uspješnosti probija na polja kao što je stomatologija i dr.[1]



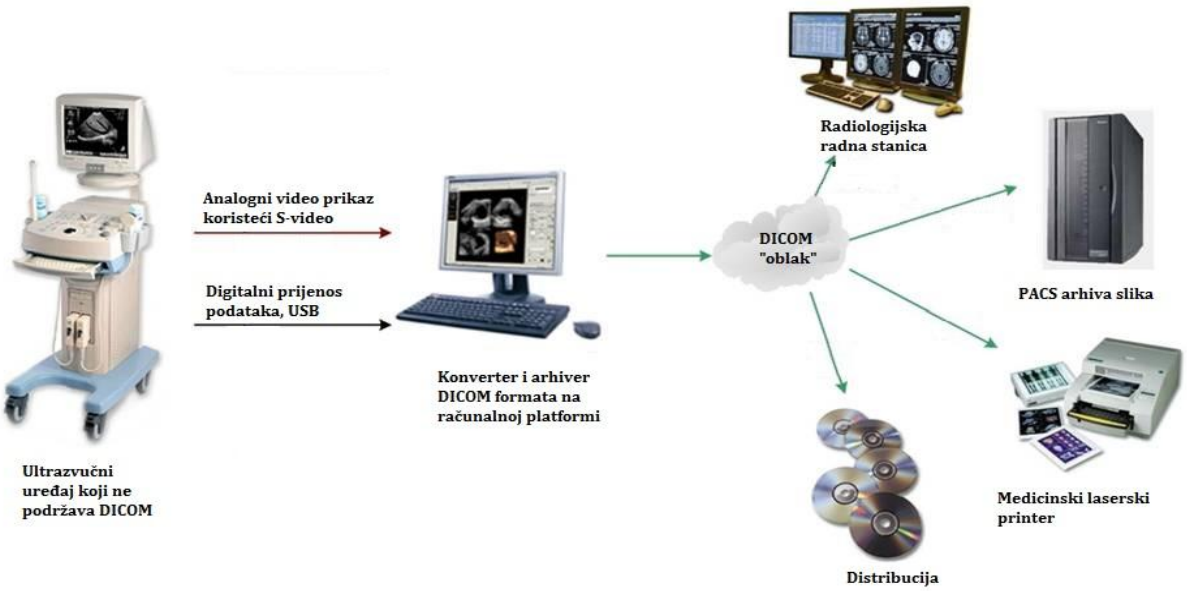
Slika 2. Primjer DICOM snimke

DICOM se razlikuje od nekih formata podataka u tome što grupira informacije u podatkovne setove. To znači da, primjerice, snimak nečije vilice sadrži i pacijentove identifikacijske podatke unutar snimka, tako da snimke međusobno nikad ne mogu biti pomiješane ili zamjenjene. To je slično kao kod, primjerice, JPEG formata koji može uz podatke o slici imati ugravirane tagove koji opisuju prikaz.[1]

DICOM podatkovni objekt sastoji se od broja atributa od kojih su neki: ime, identifikacijski broj pacijenta te podaci o broju i razmaku piksela. Jedan DICOM prikaz može imati samo jedan atribut koji opisuje broj i raznak piksela, što je za većinu primjena dovoljno. [1]

No, DICOM format može u jednom podatkovnom paketu sadržavati više slojeva omogućujući spremanje više prikaza u jedan podatkovni ovjekt. Još jedan primjer je NM podatakovni paket, gdje je NM, po definiciji, više-dimenzionalni više-okvirni prikaz. U tim slučajvima trodimenzionalni ili četverodimenzionalni podaci mogu se spremiti u jedan DICOM podatkovni objekt. Podaci o pikselima mogu se sabiti koristeći brojne standarde uključujući JPEG, JPEG Lossless, JPEG 2000 i RLE (*engl*. Run-lenght encoding). Zip kompresijom moguće je sabiti cijeli DICOM podatkovni objekt no to se rijetko koristi. Isti bazični format koristi se u svim aplikacijama uključujući mrežno korištenje.[1]

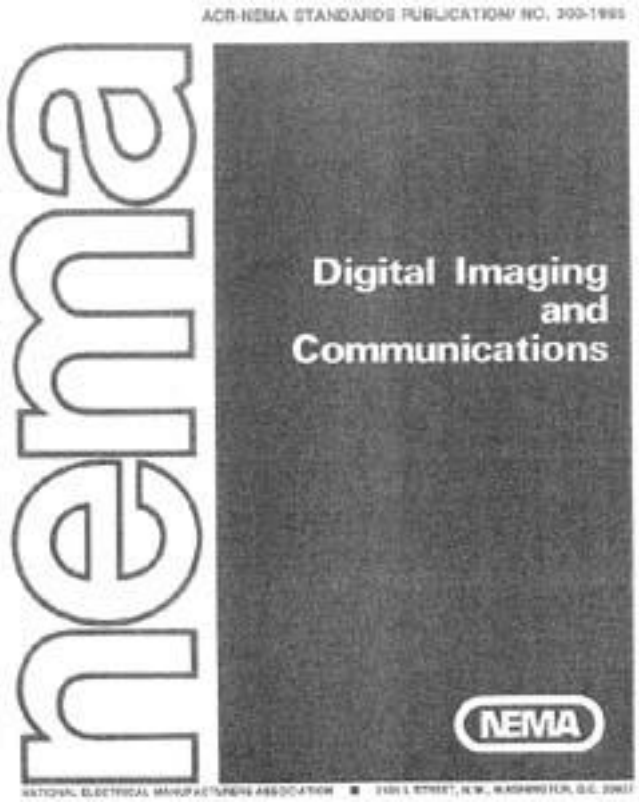
Na slici 3 prikazan je primjer DICOM „oblaka“ koji služi za mrežnu razmjenu podataka. Nakon što je format stvoren, bilo od uređaja koji podržava DICOM format ili od strane programskog konvertera, isti se može podijeliti na mrežu u tzv. oblak. Prilikom potrebe za snimkama zbog printanja, distribucije, arhiviranja ili obrade, snimke se izvlaće iz „oblaka“ putem mreže.[1]



Slika 3. Način rada DICOM oblaka [4]

4.1. Povijest DICOM formata

Prva verzija formata razvijena je na Američkom Fakultetu Radiologije (*engl*. ACR, American College of Radiology) te Nacionalne udruge električnih proizviđača, NEMA. U početku osamdesetih godina prošlog stoljeća bilo je teško za sve, osim proizvođača računalne termografije i uređaja za prikaz slike na osnovu magnetske rezonance, dekodirati slike koje su takvi uređaji proizveli. Radiolozi i medicinski stručnjaci htjeli su koristiti snimke za planiranje liječenja putem radijacije. ACR i NEMA udružili su snage te su 1983. godine formirali odbor za standarde, ACR/NEMA 300. Standard je lansiran 1985. godine. Vrlo brzo nakon izlaska standarda, postalo je jasno da su poboljšanja prijeko potrebna. Prvo izdanje bilo je prilično nejasno te je imalo nekoliko unutarnjih kontradiktornosti.[1]



Slika 4. Naslovna strana prvog izdanja ACR/NEMA standarda izdanog 1985 godine [1]

1988. godine izdana je druga verzija koja je bolje prihvaćena. Prva demonstracija druge verzije ACR/NEMA standarda održala se 1990. godine u Gorgetown sveučilištu. Šest kompanija sudjelovalo je predstavljanju, DeJarnette Reasrch Systems, General Electric Medical Systems, Merge Technologies, Siemens Medical Systems, Vortech i 3M. Komercijalna oprema koja je podupirala standard predstavljena je od strane istih prodavača 1990. godine na godišnjem skupu Radiološkog društva Sjeverne Amerike. Mnogi su ubzo shvatili da i druga verzija zahtjeva puno poboljšanja. [1]

Prva velika pošiljka ACR/NEMA tehnologije bila je 1992. godine od stran Američke vojske te ratnog zrakoplovstva kao dio projekta evaluacije medicinskih dijagnostičkih snimaka. Projekt su podrzale svi veći kompleksi za pružanje medicinske pomoći vojnicima i pripadnicima ratnog zrakoplovstva te klinike diljem SAD-a. [1]

1993. godine izdana je treća verzija standarda. Ime je tada promjenjeno u „DICOM“ nadajući se da će pod novim imenom biti prihvaćen kao standard. Definirane su nove klase usluga te je dodana mrežna podrška.

Službeno, verzija 3 je posljednja verzija, iako se standard od 1993. godine više puta obnavljao. Kada bi bila izdana nova verzija umjesto rastućeg broja verzije, u imenu bi stajala godina izdanja, npr. DICOM 2007. DICOM standard postigao je gotovo jednoglasan nivo prihvaćanja među prodavačima opreme za prikaz medicinskih podataka. [1]

Programski paket MATLAB podržava rad i manipulaciju DICOM formatom, te već ima ugrađene funkcije za čitanje takvih snimaka. Također, MATLAB je opremljen funkcijama za čitanje podataka DICOM snimaka koji se odnose na sporedne podatke, informacije o pacijentu, datum, ustanova gdje je snimanje provedeno i sl.

5. Programiranje u MATLAB-u

Matlab je *high-performance* programski jezik namjenjen za tehničke proračune. Objedinjava računanje, vizualizaciju i programiranje u lako uporabljivoj okolini u kojoj su problem i rješenje definirani poznatom matematičkom notacijom. Uobičajena je uporaba MATLAB-a za:

- matematiku i izračune,
- razvoj algoritama,
- modeliranje, simulaciju, analizu,
- analizu i obradu podataka, vizualizaciju,
- znanstvenu i inženjersku grafiku,
- razvoj aplikacija, uključujući i izgradnju GUI.[5]

U srcu Matlaba nalazi se pojam matrice, o čemu govori i samo ime Matlab koje potječe od engleskih riječi MATrix LABoratory. Matrica je jednostavan matematički objekt, pravokutna tablica brojeva, koja se prirodno javlja u najrazličitijim područjima i situacijama, dok jezgru Matlaba čini skup funkcija za jednostavno, prirodno i efikasno manipuliranje matricama. Upravo iz tog razloga MATLAB se sve više širi i u specijalizirana područja. To je multifunkcionalni programski sustav koji u jednom softverskom paketu i na jednom mjestu ujedinjuje funkcionalitete:

- numeričkih,
- simboličkih i
- grafičkih sustava.[5]

Jedna od jačih strana MATLAB-a je činjenica da njegov programski jezik omogućava izgradnju vlastitih alata za višekratnu uporabu. Možete lako sami kreirati vlastite funkcije i programe (poznate kao *M-datoteke*) u kodu MATLAB-a. Skup specijaliziranih M-datoteka za rad na određenoj klasi problema naziva se *Toolbox*. [5]

S MATLAB-om dolazi nekoliko *Toolbox-ova*koji su i više od kolekcije korisnih funkcija; oni predstavljaju rezultate istraživanja vrhunskih stručnjaka iz područja upravljanja, obrade signala, identifikacije procesa, i drugih. Dakle uz pomoć MATLAB-a možete sami razviti nove ili adaptirati postojeće Toolbox-ove za rješavanje određenih problema.Neke od prednosti MATLAB-a spram nekih drugih programskih jezika su:

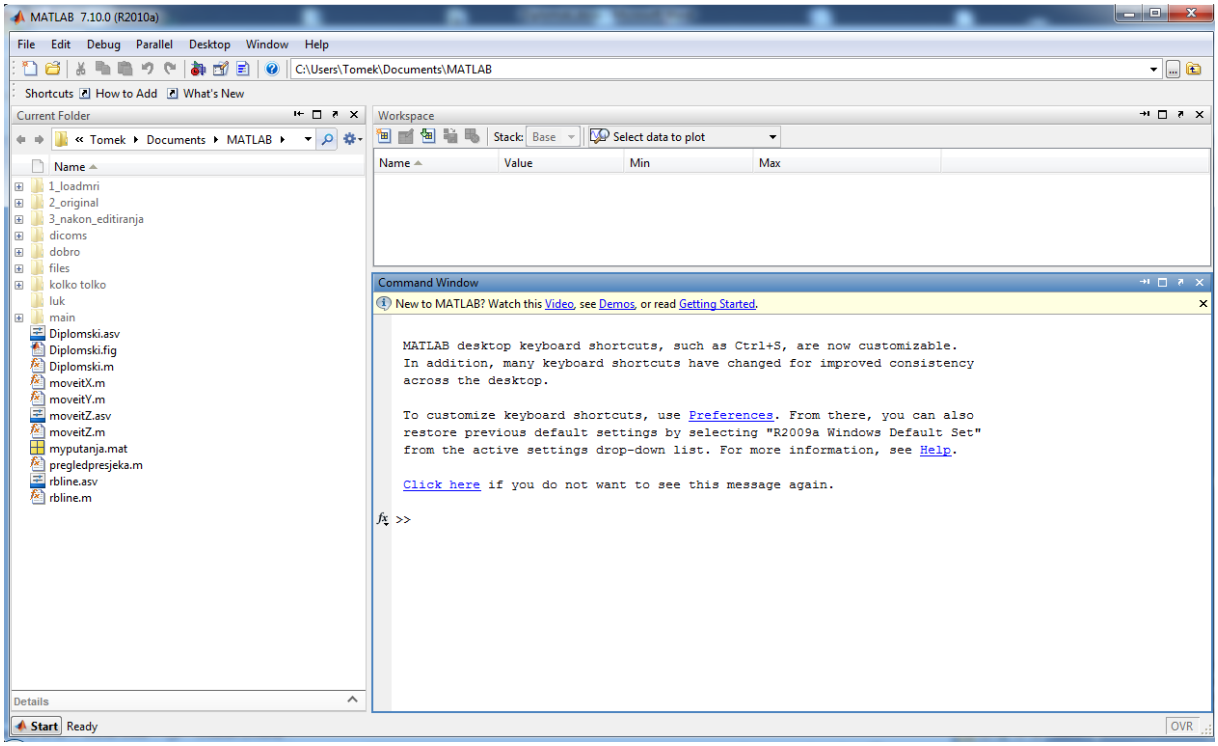
- interaktivno sučelje,
- brzo i lako programiranje,
- ugrađeni grafički sustav omogućuje jednostavnu, bržu i kvalitetnu vizualizaciju,
- programi pisani Matlabovim jezikom su obične tekstualne datoteke i stoga su potpuno prenosive između različitih operacijskih sustava/ platformi,
- mnogobrojni dodatni paketi *(toolbox-i)* za razna specijalna područja,
- mnogoborojne m-datoteke i čitavi paketi koje autori, ujedno i korisnici, stavljaju na slobodno raspolaganje putem Interneta.[5]

Osnovni elementi programskog paketa MATLAB su:

- **Razvojna okolina:** Skup alata za lakšu uporabu *Matlab*a i njegovih funkcija. Mnogi od ovih alata (u verziji 6) su realizirani u grafičkom sučelju. To su *Matlab* desktop, komandni prozor (*engl.* Command Window), povijest naredbi (*engl.* command history), editor i debuger, te preglednici help-a, radnog prostora (*engl.* Workplace), datoteka te path-a.
- **Biblioteka matematičkih funkcija:** Ogromna kolekcija računalnih algoritama.
- **Programski jezik:** *Matlab* programski jezik je jezik visokog stupnja matrično orjentiran, s naredbama uvjetnih struktura, funkcija, struktuiranim podacima, ulazomizlazom te nekim svojstvima objektno-orjentiranog programiranja. Ovaj programski jezik omogućava programiranje na nižoj razini (kao npr. za potrebe studenata) ali i za kompleksnije programe većih razmjera.
- **Grafički alat:** *Matlab* raspolaže velikim mogućnostima za grafički prikaz podataka, vektora i matrica, kao i notaciju i printanje tih dijagrama. Postoje funkcije visokog stupnja za 2D i 3D vizualizaciju podataka, obradu slike, animaciju. Također postoje i funkcije za izgradnju grafičkih sučelja za vaše *Matlab* aplikacije.
- **Sučelje programskih aplikacija:** Biblioteka za interakciju s drugim jezicima.[5]

5.1. Radni prostor MATLAB-a

Rad u MATLAB–u odvija se putem radnog prozora, a unutar radnog prostora. Radni prozor služi za unos naredbi, te ispis rezultata i najvažniji je dio korisničkog sučelja. Radni prostor zamišljamo kao dio radne memorije dodijeljen MATLAB-u prilikom pokretanja, koji sadrži korisnikove varijable. Moglo bi se reći da se kroz radni prozor "gleda" u radni prostor.



Slika 5. Radni prostor MATLAB-a

Osnovni prozor Matlaba sadrži izbornik, redak s alatima (*toolbar*) i pet mogućih podprozora. Odgovarajući prozor aktivira se u izborniku *Desktop*. *Launch Pad* prikazuje osnovnu strukturu MATLAB-a. *Workspace* je prozor koji omogućuje uvid u memorijski prostor, odnosno u tekući direktorij. Podprozor *Command History* sadrži popis svih do sada upotrebljenih naredbi. U *Command Window* se upisuju naredbe MATLAB-a iz kojih se dobivaju numerički rezultati.

Sa lijeve strane slike 5 vide se datoteke koje se nalaze u aktivnom folderu MATLAB-a iz kojeg se učitavaju sve ne integrirane funkcije. Primjerice ako se želi pozvati neka odprije kreirana funkcija, ona mora biti u aktivnom folderu, inače MATLAB ju neće registrirati i neće ju biti moguće pozvati.

Redak teksta upućuje na *Help* , odnosno pomoć koju korisnik može zatražiti od programa. Znak >> predstavlja prompt MATLAB-a. Označava da je Matlab spreman za prihvat naredbi od korisnika. Iza prompta moguće je upisivati naredbe Matlaba, pokretati funkcije i izvršavati matematičke operacije.

U traženju pomoći pri definiranju i unosu naloga treba kliknuti na nalog *Help/MATLAB Help* u traci s nalozima te u otvorenom dijalognom prozoru u kartici *Search* upisati nalog za koji se pomoć traži. MATLAB je i okružje i programski jezik. Jedna od jačih strana Matlab-a je činjenica da njegov programski jezik omogućava izgradnju vlastitih alata za višekratnu uporabu. Možete lako sami kreirati vlastite funkcije i programe (poznate kao *m-datoteke*) u kodu MATLAB-a. MATLAB posjeduje niz *demo* funkcija za prezentaciju mogućnosti Matlab-a. Postoje funkcije visokog stupnja za 2D i 3D vizualizaciju podataka, obradu slike, animaciju. Također postoje i funkcije za izgradnju grafičkih sučelja za vaše MATLAB aplikacije.

5.2. Struktura i varijable

MATLAB koristi slijedeće elementarne operacije i funkcije: aritmetičke, logičke i operacije izmjene podataka te matematičke i grafičke funkcije. One se izvode na objektima određenog tipa podataka, a to su matrice. Iz tog se razloga matricama ovdje pridodaje naziv MATLAB varijable. One mogu sadržavati realne i kompleksne brojeve te ASCII znakove. Funkcije rade nad varijablama koje se trenutno nalaze u radnom prostoru ili kreiraju nove varijable. Po izvršenju funkcije sve varijable ostaju u radnom prostoru i može ih se pregledati, crtati, te dalje koristiti. [6]

Cjelokupni rad s MATLAB-om zasniva se na radu s varijablama. Na definirane ulazne varijable primjenjuju se matematičke operacije i funkcije, a kao rezultat dobiju se izlazne varijable. Varijabla se definira i unosi u komandni prozor preko tipkovnice uz slobodni izbor njenog naziva, kao na primjer:

```
>>x=3.25

x =

3.2500
```

Format ispisa rezultata ostvaruje se preko podešavanja odgovarajuće postavke iz menija *File/Preferences/Command Window/numeric format*; ili iz komandnog prozora naredbom *format*. Temeljni je format short koji ispisuje 4 značajna decimalna mjesta. To se može promijeniti odabirom opcije long ili naredbom format long. Upisivanje matrica prikazano je slijedećim primjerom:

```
>>[2,-4,5;-8,3,-2]

ans =

 2 -4  5
-8  3 -2
```

Ne upiše li se naziv varijable MATLAB sam dodaje naziv *ans* (odgovor).

Iz ovih primjera proizlazi da su unosi jednog reda matrice međusobno odvojeni praznim mjestima, a jednog stupca s točka – zarezom. Točka – zarez na kraju naredbe označava da se rezultat ne ispisuje na ekranu. Pored toga ; služi za razdvajanje više naredbi u jednom redu. Ukoliko nam je naredba predugačka za jedan red dodavanjem na kraju tog reda ... ista se nastavlja u sljedećem redu. [6]

MATLAB varijable mogu sadržavati realne i kompleksne brojeve te ASCII znakove. Varijable u Matlabu su matrice, različitog sadržaja i dimenzije. Skalarne se veličine tretiraju kao matrice 1*1, dok su vektori stupčane ili redne matrice.

Kao i kod svih računalnih jezika postoje neka od osnovnih pravila kod imenovanja varijabli i datoteka:

- Potrebno je razlikovati uporabu malih i velikih znakova *(case sensitive),*
- Maksimalni broj znakova,
- Prvi znak mora biti slovo.[6]

MATLAB posjeduje i specijalne (simboličke) varijable (pi, eps, ans…) čiji su nazivi rezervirani. Simboličke varijable sadrže simbole koje se interpretiraju numerički. Varijable Matlaba se mogu podijeliti u više skupina (prema sadržaju elemenata matrice, prema dohvatu, prema izvoru nastanka) a njihova svojstva ovise o pripadnosti pojedinim skupinama. Sadrže li elementi matrice realne brojeve i varijabla se može nazvati realna. Kompleksne varijable sadrže kompleksne brojeve. [6]

Prema vidljivosti varijable možemo podijeliti na lokalne i globalne.Ukoliko je potrebno određene varijable primijeniti u nekoliko funkcija, a želimo izbjeći njihovo pojavljivanje u popisu ulaznih varijabli tih funkcija, dovoljno je takve varijable deklarirati kao globalne. Globalne su varijable one varijable koje su vidljive iz više funkcija MATLAB-a. Postoji li definirana varijabla u funkciji MATLAB-a, nakon izvršenja funkcije u Workspacu varijabla neće postojati. Vidljiva je samo u funkciji gdje je definirana. Takve varijable tretiramo kao lokalne. Varijabla postaje globalna tako da se deklarira naredbom:

```
>>global ime_varijable
```

Prema izvoru nastanka varijable možemo podijeliti na interne i eksterne varijable.
Interne su varijable one varijable koje definira sam Matlab kao što su *eps, realmin, realmax, pi, inf,* itd. [6]

Bilo koji od ovih varijabli interpretiraju se jednostavnim ukucavanjem naziva u komadni prozor iza znaka prompt. Eksterne varijable su varijable definirane od korisnika ili varijable nastale kao rezultat matematičkim operacijama i funkcija izvedenih u Matlabu. Definirane varijable moguće je vidjeti u tzv. *Workspace-u* (pregledniku radnog prostora) ili u komadnom prozoru primjenom naredbe *who (*ispis varijabli) i *whos (*detaljni ispis varijabli).[6]

Samim nalogom *clear* bez dodatnih opcija izbrisali bi sve varijable iz radnog prostora. Ista mogućnost nam stoji na raspolaganju s preglednikom radnog prostora - pritiskom desne tipke miša možemo odabrati *Delete* za brisanje odabrane varijable (ili ikonicu za brisanje) te *Clear Workplace* za brisanje svih varijabli iz radnog prostora. Sadržaj radnog prostora možemo spremiti s binarnim formatom u željenu datoteku *ime*.mat i to na nekoliko načina:

- iz menija *File/Save Workspace as,*
- iz menija *File/Import Data,*
- iz preglednika radnog prostora - pritiskom desne tipke miša odabirom *Save Workplace as,*
- naredbom u komandnom prozoru *save* ime.[6]

Dijalogni prozor *File/Import Data* sadrži popis datoteka različitih formata. Klikom na bilo koju datoteku otvara se prozor *Import Wizard*. Na taj način se mogu izvesti odgovarajuće izmjene u datoteci. Matlab koristi više raznih vrsta datoteka i one se prepoznaju po nastavku unutar imena. Najčešći korišteni tipovi su:

- *ime.m* - je tekstualna datoteka koja sadrži program pisan u Matlabu,
- *ime.mat* - je binarna datoteka, tj. standardni Matlabov format za spremanje varijabli i njihovih vrijednosti.
- *ime.fig* - je binarna datoteka, tj. standardni format Matlaba za spremanje slika.[6]

5.3. Grafičke funkcije

Osim matričnog koncepta, druga glavna točka MATLAB-a je izvrstan grafički podsustav. Interne grafičke funkcije koje su odmah dostupne omogućavaju relativno lako crtanje slikovnih prikaza. S druge pak strane moguće je detaljno upravljanje svim aspektima prikaza kao što su veličina i položaj slike na papiru, položaj označavanja osi itd.

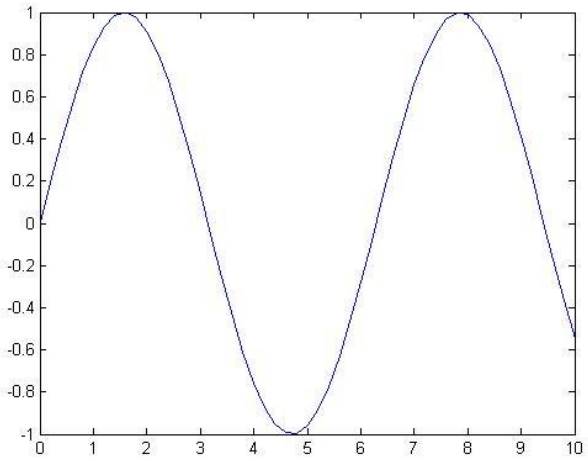
Za dvodimenzionalni prikaz potrebna su dva vektora iste duljine a za trodimenzionalni tri. Kad se želi istovremeni prikaz zavisnosti više funkcija o jednoj varijabli, onda je varijabla vektor a funkcije se prikazuju matricom. Uz to se linije grafa mogu prikazati na različite načine. Najvažniju funkciju ima nalog *plot*, a unosom naloga *help plot* dobivaju se sve potrebne informacije o ovoj funkciji. U svrhu grafičkog prikaza neka se varijabla *x* može, kao vektor odnosno jednoredna matrica, definirati i unijeti i na slijedeći način:

```
>>t=(0:0.2:1)

t=

0  2  4  6  8  10
```

gdje je *t* vremenski vektor, prva znamenka u zagradi označuje prvu vrijednost varijable, srednja razmak pojedinačnih vrijednosti a zadnja konačnu vrijednost. One su međusobno odvojene dvotočkom. Želimo li prikazati graf *sin t*, onda unos naloga ima izgled:

```
>>t=(0:0.2:10);

>>s=sin(t);

>>plot(t,s)
```

Znakom „;“ govorimo MATLAB-u da naredbe koje prethode tom znaku kalkulira u sebi te ih nije potrebno prikazivati na ekranu. Ovoga puta povećano je uzorkovanje tako da bi graf ispao glađi. Nakon unosa koda dobije se prikaz kao na slici 6.[6]



Slika 6. Primjer ispisa sinusoide pomoću Matlaba

U grafovima se jednostavnim nalozima, baziranim na uputama u već spomenutom nalogu *help plot,* mogu mijenjati boje i vrste linija. U cilju dokumentiranja grafova nužno je uz to upisati oznake osi i naziv dijagrama, ucrtati mrežu pomoćnih linija te povećati pojedine isječke dijagrama.[6]

Za opise služe nalozi *xlabel, ylabel i title*, za mrežu pomoćnih linija nalog *grid*, a za povećanje isječka nalozi *zoom* i *axis* u kojima se mora navesti x,y područje koje se želi povećati. [6]

Osim funkcije *plot* postoje i funkcije *semilogy, semilogx, loglog, plotyy* za crtanje u logaritamskom mjerilu, te mnoge druge funkcije za crtanje dvodimenzionalnih prikaza. Sve ove funkcije imaju različitu sintaksu za različite potrebe korisnika. Više o tim funkcijama možete doznati pozivanjem naloga *help ime_funkcije*.[6]

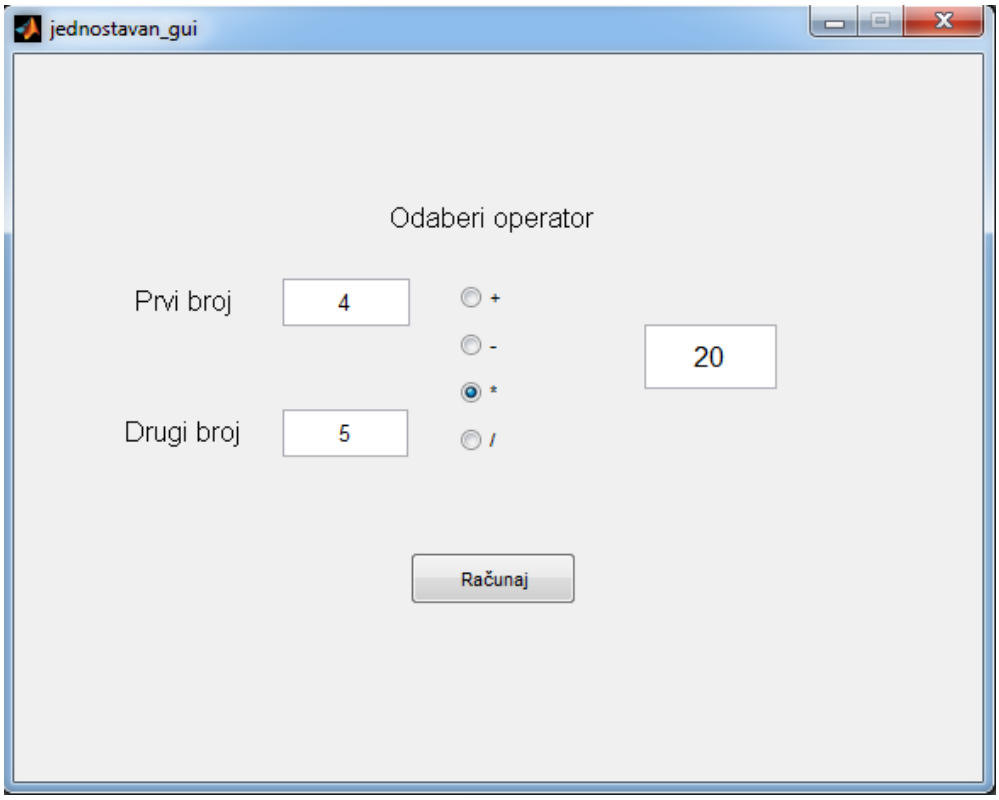
Naknadnim pozivanjem neke grafičke funkcije poput *plot* postojeća slika, će u pravilu, biti obrisana i zamijenjena novom. Ako se pak želi crtati preko postojeće slike treba koristiti naredbu *hold on*. Suprotan učinak ima naredba *hold off*.[6]

5.4. Grafičko korisničko sučelje

Grafičko korisničko sučelje (*engl.* Graphical User Interface) je grafički prikaz, u jednom ili više prozora, palete komponenata koje omogućuju korisniku izvršavanje interaktivnih zadataka. Korisnik grafičkog sučelja ne mora kreirati .m skripte ili unositi kodove da bi izvršio određene zadatke. Za razliku od kodiranja programa za izvršenje određenih zadataka, korisniku GUI-a nije potreno razumjevanje detalja i kodova koji su omogućili da zadatak bude izvršen.

GUI komponente uključuju menije, alatne trake, raznovrsne gumbe, polja za odabir, padajuće menije, klizače i ostale komponente koje služe korisniku za definiranje i uspješno provođenje zadatka.

GUI također može vršiti različite kalkulacije, pisati i čitati podatke, komunicirati sa ostalim GUI-ima te ispisivati podatke kao tablice ili dijagrame. Na slici 7 prikazan je jednostavan GUI kojega čak i laik može izraditi u nekoliko minuta.

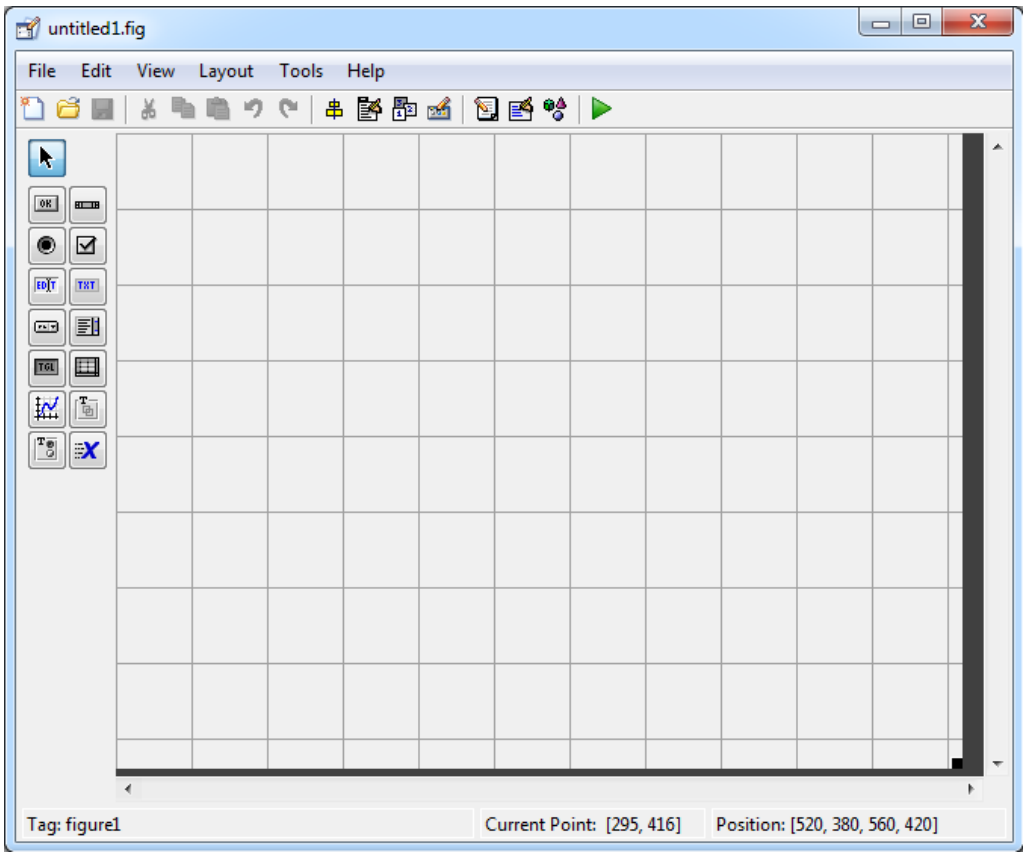


Slika 7. Izgled jednostavnog grafičkog korisničkog sučelja

Na slici je prikazan jednostavan GUI koji ima mogućnost upisa dvaju brojeva te odabirom funkcije zbraja, oduzima, množi ili dijeli brojeve pritiskom na tipku „Računaj“. Rezultat ispisuje u prozor koji se nalazi u desnom dijelu sučelja. Iza tako napravljenog GUI-a sakrivaju se tzv. „pozivne“ funkcije. Pozivna funkcija se aktivira po, unaprijed određenoj, naredbi korisnika, bilo to pritiskom na tipku ili nekom drugom akcijom korisnika.

Na taj se način jednim klikom mogu aktivirati brojne funkcije koje se nalaze u .m datoteci pripadajućeg GUI sučelja. Unutar GUI-a također se mogu uključivati i koordinatne osi sa ucrtanim podacima.

GUI se može stvoriti programski ili kroz integrirano Matlabovo sučelje „Guide“. Programski znači da sve komponente GUI-a stvaramo prvenstveno unosom koda u pripadajuću .m datoteku. Programski unos je namjenjen za korisnike sa više iskustva u radu Matlabom. Na taj način korisnik ima potpunu kontrolu izrade GUI-a te izbjegava suvišne redove koda koji se mogu pojaviti korištenjem *Guidea*. Korištenjem *Guidea* izrada GUI-a je jednostavnija i brža te je prvi odabir ukoliko korisnik nije iskusan u programiranju Matlabom. Unosom „guide“ u radni prozor Matlaba otvara se sučelje za izradu GUI-a. Nakon što se odabere GUI predložak otvara se prozor kao na slici 8.



Slika 8. Predložak za izradu grafičkog korisničkog sučelja

Sa lijeve strane ponuđene su komponente. Ovdje se nalaze tipke, polja za unos, statički tekst, tipke za odbir, klizači, padajući meniji, koordinatni sustavi za umetanje i dr. Ukoliko je potrebno da korisnikov GUI ima neku od komponenata to se postiže „drag & drop“ akcijom na bilo koju od komponenata.

Desnim klikom miša na umetnutu komponentu te odabirom „*Property Inspector*“ mogu se mjenjati različita svojstva komponenata kao što su pozadina, boja, tekst, veličina, font i sl.

Ukoliko je potrebno da se pritiskom na tipku izvrši određena funkcija, potrebno je funkciju upisati unutar pozivne funkcije te tipke. To se čini tako tako da se desnom tipkom miša klikne na željenu tipku te se odabere *„View Callbacks -> Callback“.* GUI će automatski otvoriti pripadajući .m file na mjestu pozivne funkcije odabrane tipke, gdje je potrebno upisati funkciju koja se pritiskom na tu tipku izvršava.

Pri izradi programskog sučelja treba imati na umu tko će se programom služiti. Primjerice, ako se program radi za djecu, sadržavati će, umjesto pisanih riječi, jasno obojane slike na koje se može klikati mišem. S druge strane ovakav pristup ne bi bio primjeren ukoliko je program namjenjen ozbiljnijem korisniku ili publici. Najlakše je grafičko sučelje predočiti ako zamislimo moderno računalo. Sve zadatke koje korisnici obavljaju na računalima zapravo odrađuju preko grafičkog korisničkog sučelja dok se funkcije vrte u pozadini. Pomoću grafičkog sučelja uporaba današnjih računala je mnogo jednostavnija nego u doba DOS operativnog sustava koji je bio prilično negostoljubljiv prema novim korisnicima računala. Većina današnjih operativnih sustava ima mogućnost uporabe grafičkog sučelja, dakle kursora, ikona, prozora i drugih elemenata.

6. Manipuliranje DICOM snimkama

6.1. Izrada realnog 3D prikaza

Primjena robota i strojeva u medicinske svrhe postala je neizbježna. Operacije su sigurnije i brže uz pomoć pametnih strojeva. Pomoću robota mogući su zahvati koji nisu bili sigurni ako bi ih izvodio čovjek. Jedna od takvih primjena je korištenje robota za neutraliziranje živca u mozgu koji je uzrok epileptičnim napadima. Tehnologija se zasniva na usklađivanju koordinata robota sa koordinatama živca. Tome prethodi niz postupaka i akcija. Prije svega valja skenirati lubanju i mozak magnetskom rezonancom te iz dobivenih slika stvoriti trodimenzionalni prikaz ljudske lubanje. Mora biti moguće i manipuliranje tim prikazom u svrhu dobivanja rezova iz drugih kuteva. Radi lakšeg shvaćanja postupka izrade prikaza i programa općenito biti će prikazani isječci programskog koda. Tekst koji slijedi iza znaka „%“ je komentar te služi programerima za lakše snalaženje u programskom kodu.

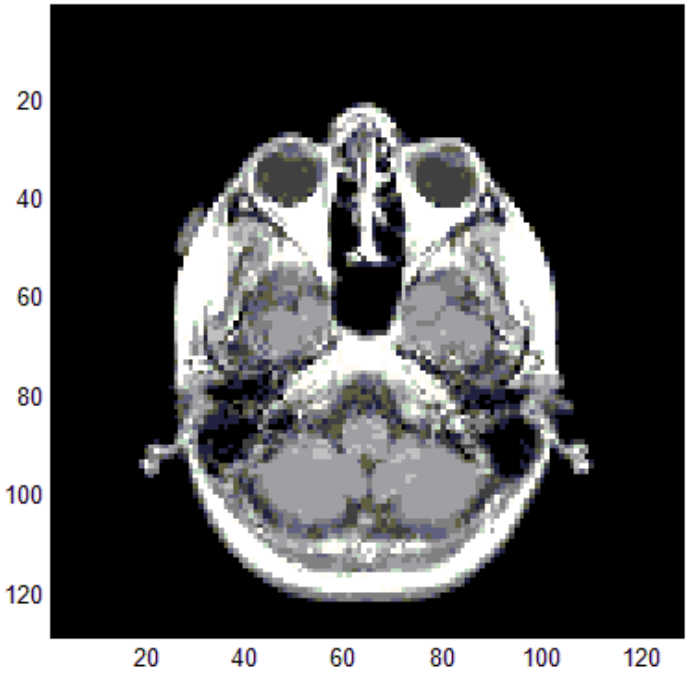
Prije svega valjalo se upoznati sa MATLAB-om i načinom na koji interpretira Dicom format snimaka. Prilikom instalacije MATLAB-a, instaliraju se mnogi paketi i dodaci. Tako, u MATLAB-u, već postoji niz MR snimaka, točnije njih 27. Pozivamo ih naredbom:

```
>> load mri;
```

matlab tada učitava snimke i sprema ih u obliku trodimenzionalne matrice [127;127;27]. Što znači 27 snimaka rezolucije 127 x 127 piksela. Snimke su 8 bitne. Za prikaz jednog presjeka potrebno je upisati slijedeće naredbe:

```
D = squeeze (D);    %funkcija squeeze uklanja jedinične dimenzije matrice
num = 4;            %postavljanje željene slike

image(D(:,:,num))   %prikaz 4. slike iz matrice D
axis image;         %poravnavanje osi: služi da npr elipsa izgleda kao
                    %elipsa a ne kao kružnica
colormap(map);      %postavljanje palete boja

x=xlim;             %dobavljanje limita x osi
y=ylim;             %dobavljanje limita y osi
```

Na taj način otvara se zaseban prozor u matlabu i ispisuje četvrti po redu snimak od integrirane palete snimaka u Matlabu. Snimak je prikazan na slici 9.



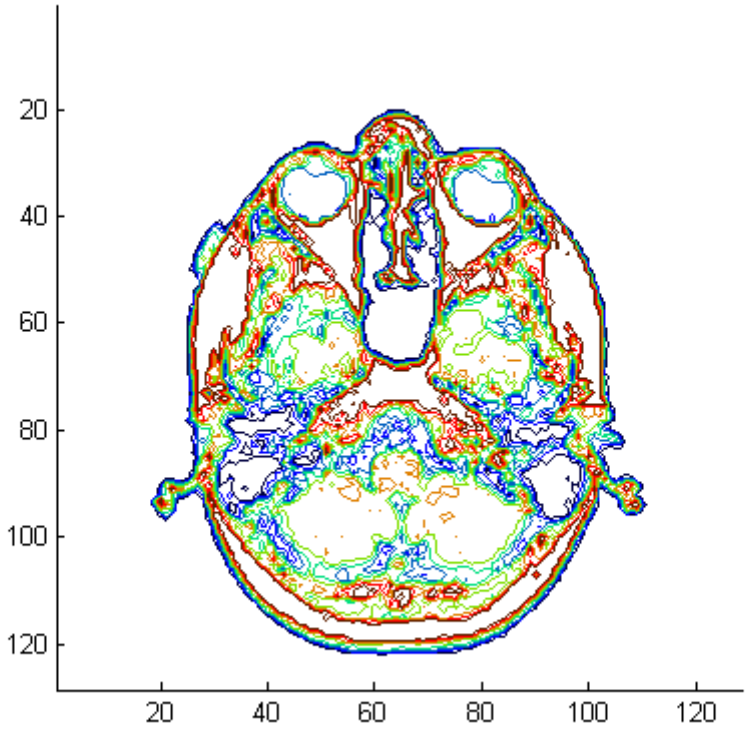
Slika 9. Otvaranje DICOM snimaka u Matlabu

Ideja je za početak bila napraviti postupak vizualizacije već ugrađenih snimaka za koje je sigurno da su ispravne te bi se njima trebalo lako manipulirati. Kada je postupak napravljen, umjesto tih snimaka stavile bi se originalne snimke snimljene u KBC Dubrava. Realni 3D prikaz ostvariti ćemo „slaganjem" jednog presjeka na drugi te tako stvoriti i treću dimenziju koja nedostaje. Da bi se kao krajnji rezultat dobio 3D prikaz od 27 presjeka potrebno je najprije svakoj slici odrediti rubne konture: To se postiže slijedećim kodom:

```
cm = brighten(jet(length(map)),-.5); %brighten funkcija smanjuje svijetlost
                                     %slike kako
                                     %bi se bolje vidjeli detalji. Jet je
                                     %set boja sličan HSV-u.
figure('Colormap',cm)                %stvara novi figure sa paletom boja
                                     %dužine prethodne slike
contourslice(D,[],[],num)            %naredba contourslice povlači konture
                                     %(obrube) na
                                     %volumnim plohama (slices)
axis ij                              %axis ij postavlja matlab u matrični
                                     %oblik osi.
                                     %Početak koordinatnog sustava je u
                                     %gornjem lijevom kutu. I os je
```

```
                                    %vertikalna i numerirana je od vrha
                                    %prema dnu. J os je vodoravna i
                                    %numerirana je od lijevo prema desno
xlim(x);                            %postavljanje limita x osi
ylim(y);                            %postavljanje limita y osi
daspect([1,1,1]);                   %postavlja omjer proporcija podataka
```

Nakon čega se dobije slijedeća slika:



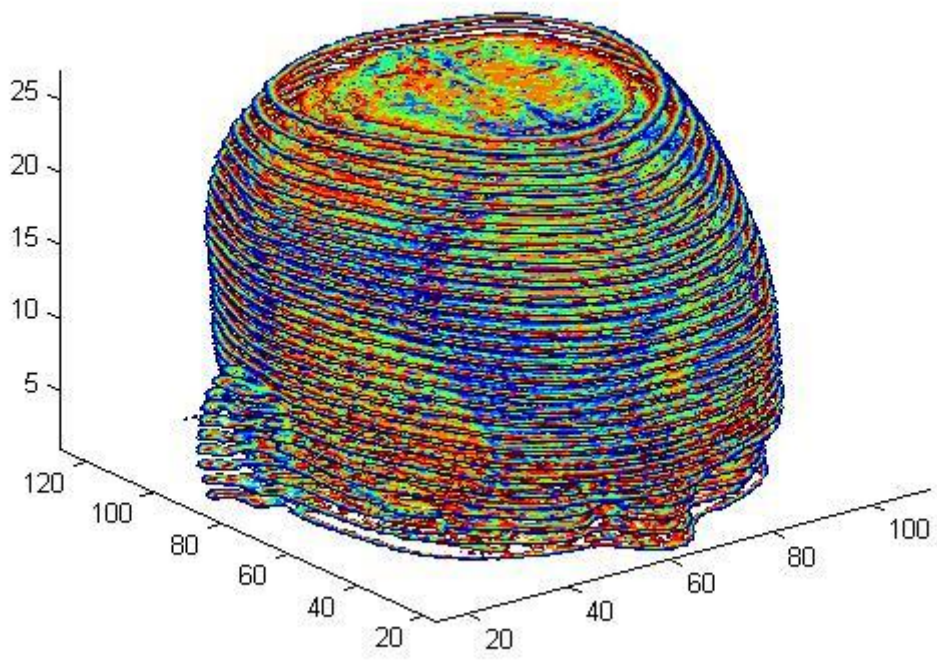
Slika 10. Određivanje rubnih kontura presjeka

Sve snimke su u crno bijeloj kombinaciji. To znači da svaki piksel na slici može poprimiti jednu vrijednosti između 0 i 255, gdje je 0 potpuno crno a 255 potpuno bijelo. Upravo to je vrlo bitno kod stvaranja kontura na presjeku. Crvenim rubom označuje se područje gdje je raspon vrijednosti jednog piksela do drugog bio najveći, dok plavom tamo gdje je taj raspon bio najmanji. Područja koja su bijle boje označavaju područja na kojima nije bilo razlike u intenzitetu piksela. To je prvi bitan korak u stvaranju 3D realnog prikaza ljudske glave načinjenog od niza vodoravnih snimaka.

Slijedeće, treba poslagati presjeke sa konturama jedan na drugi. To se postiže kodom kako slijedi:

```
figure('Colormap',cm)                          %postavljenje 3. slike
contourslice(D,[],[],[1,13,18,20,25],8);       %postavljanje 4 presjeka jedan
                                               %iznad drugog
                                               %sa jasno izraženim obrubima
view(3);                                       %postavlja zadani 3D prikaz
axis tight                                     %axis tihgt postavlja granice
                                               %osi uz granice podataka
```

Pokretanjem gore definiranog koda otvara se zaseban prozor koji prikazuje poslagane presjeke jedan na drugi kao na slici 11.

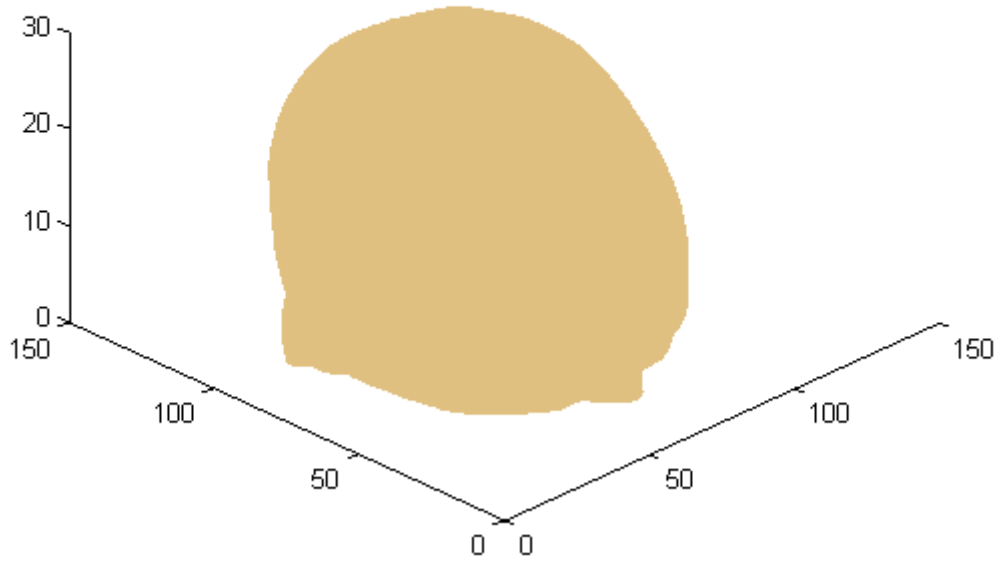


Slika 11. Slaganje rubnih kontura presjeka vertikalno jedan na drugi

Na slici vidimo poslagane presjeke jedan na drugi sa jasno definiranim konturama kao na primjeru jednog presjeka. To je prvi veći korak ka stvaranju realnog 3D prikaza ljudske glave. Čak i ovakav prikaz možemo rotirati u 3D prostoru te nam daje početni uvid u strukturu pacijentove glave. Slijedeći korak koji nas dijeli od realnog 3D prikaza ljudske glave sastoji se od dva dijela. Prvi dio koda zaglađuje površinu stvorenu presjecima s konturama, a drugi stvara tzv. izopovršinu oko presjeka. Izopovršina je, zapravo, plašt koji okružuje krajnje točke presjeka i spaja ih ucjelinu. Da bi prikaz bio još realniji dodaje se boja što sličnija boji kože:

```
figure('Colormap',map)                  %postavljanje 4. prikaza
Ds = smooth3(D);                        %naredba smooth3 zaglađuje površinu
hiso = patch(isosurface(Ds,5),...       %naredba patch stvara zakrpe između
                                        %slojeva dok naredba isosurface stvara

                                        %3D pokrov kok rubnih točaka.
    'FaceColor',[1,.75,.65],...         %postavlja boju površine plašta
    'EdgeColor','none');                %postavljanje boje ruba
isonormals(Ds,hiso)                     %naredba isonormals stvara normale od
                                        %vrhova izopovršina
                                        %služi za poboljšanje kvalitete slike
```

Generira se slijedeći prikaz:



Slika 12. Povezivanje kontura izopovršinom

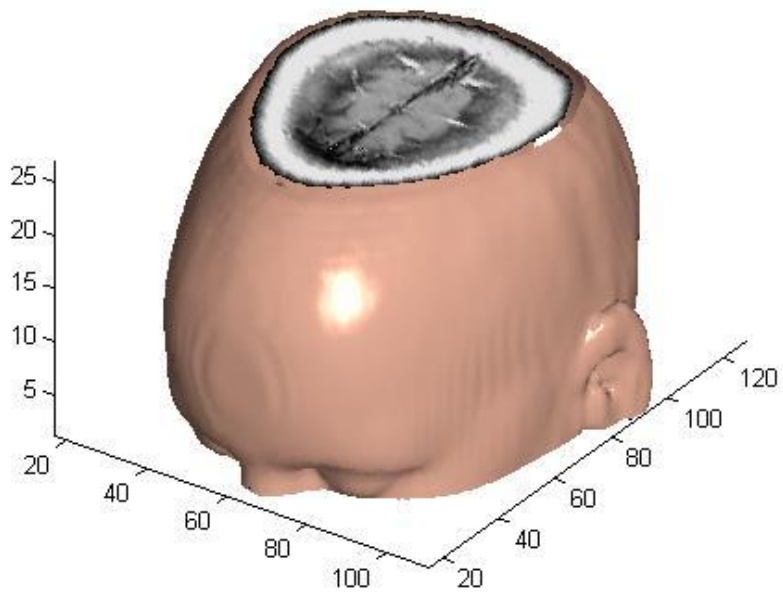
Prikaz koji se generirao također se može rotirati u prostoru, no nije dovršen te je dosta nejasan. Da bi smo dobili jasniji i realniji prikaz, izopovršina se popunjava te se dodaju razni efekti kao što su zaglađenje površine te rasvijeta:

```
%% POPUNJAVANJE IZOPOVRŠINE

hcap = patch(isocaps(D,5),...           %naredba isocaps izračunava geometriju
                                        %izopovršine
```

```
    'FaceColor','interp',...          %interp naredba reorganizira podatke
                                      %na višoj stopi(frekvenciji) koristeći
                                      %niskopropusnu interpolaciju
    'EdgeColor','none');

%% DEFINIRANJE SLIKE

view(35,30)                           %postavlja početni kut gledanja 3D
                                      %slike
                                      %prvi broj je horizontalna rotacija
                                      %ili azimut, a drugi broj je
                                      %vertikalni uspon (oba u stupnjevima)
axis tight                            %axis tihgt postavlja granice osi uz
                                      %granice
                                      %podataka
daspect([1,1,.4])                     %postavlja omjer proporcija podataka


%% POSTAVLJANJE RASVJETE I ZAGLAĐIVANJE SLIKE


lightangle(45,30);                    %postavlja poziciju umjetnog svijetla
                                      %prvi broj je horizontalna rotacija
                                      %ili azimut, a drugi broj je
                                      %vertikalni uspon (oba u stupnjevima)
set(gcf,'Renderer','zbuffer'); lighting phong
                                      %naredba gcf direktno pristupa
                                      %svojstvima trenutnog prikaza
                                      %lightning pong postavlja reflekciju
                                      %slike na prihvatljiv nivo. Dobro kod
                                      %prikaza zakrivljenih površina.
set(hcap,'AmbientStrength',.6)        %AmbientStrenght je svojstvo izokapse
                                      %da posvijeti
                                      %boje slike bez utjecaja na
                                      %izopovršinu

set(hiso,'SpecularColorReflectance',0,'SpecularExponent',50)

                                      %SpecularColorReflectance učini da
                                      %spektar reflektiranih boja bude
                                      %sličniji
                                      %spektru boja izopovršine, a
                                      %SpecularExponent služi za smanjivanje
                                      %spektra boje točke.
```

Time se zaključuje realni 3D prikaz ljudske glave. Rezultat gornjeg koda je prilično stvaran prikaz pacijentove glave, odnsno dijela glave koji je snimljen MR uređajem.

Nakon što se izvrši gore prikazani kod u zasebnom prozoru pojavljuje se slijedeća slika:



Slika 13. Popunjavanje izopovršine te definiranje slike

6.2. Izrada realnog prikaza originalnih snimaka

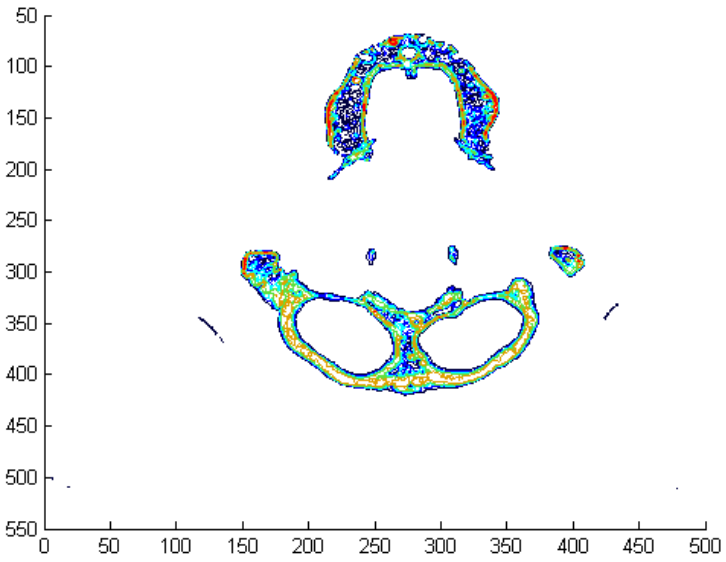
Budući da kod funkcionira za integrirane Matlabove snimke, bilo bi logično da postupak mora biti isti i za drugačiji set Dicom snimaka. Koriste se presjeci izrađeni u KBC Dubrava, u Zagrebu. Otvaranje zasebne dicom slike ostvaruje se slijedećim kodom:

```
>> I=dicomread('ime_slike.dcm');

>> imtool(I,'DisplayRange',[])
```

te se u zasebnom prozoru ispisuje snimak kao na slici 14:

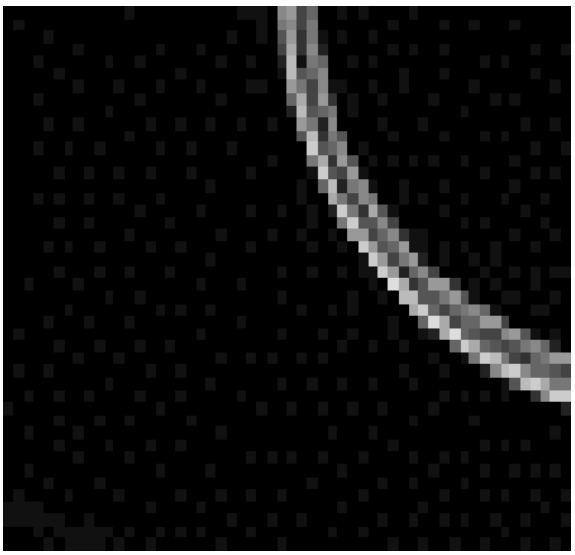
Slika 14. Čitanje originalne DICOM snimke u Matlabu

Na prvi pogled snimka je slična gornjem primjeru, osim što je veće rezolucije (512x512) te je 16- bitna. Logično bi bilo da što je slika kvalitetnija i generirani prikaz biti će kvalitetniji. Pokazalo se da je to samo dijelomično točno. Već pri postavljanju kontura na presjek javlja se prvi problem.



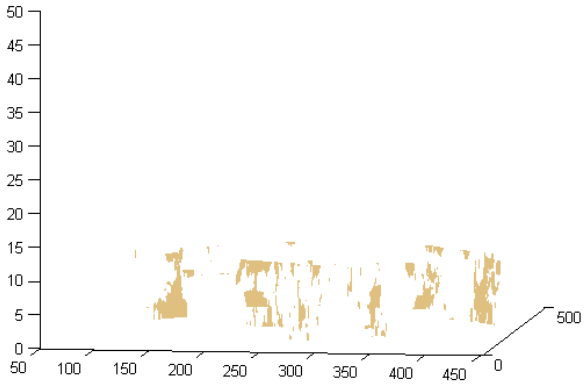
Slika 15. Pojava šuma prilikom određivanja rubnih kontura

Sa slike 15 jasno se vidi da, osim dobro generiranih kontura, javljaju se konture koje nisu dio željenih podataka. Naime, javlja se šum. Dicom snimke na prvi pogled su crno bijele slike sa jasno izraženim podacima o strukturi ljudske glave i mozga. No ako dovoljno povećamo jednu od slika, uočava se da na snimkaka ima dosta šuma koji spriječava program da jasno označi konture na presjeku.



Slika 16. Pojava šuma na originalnim DICOM snimkama

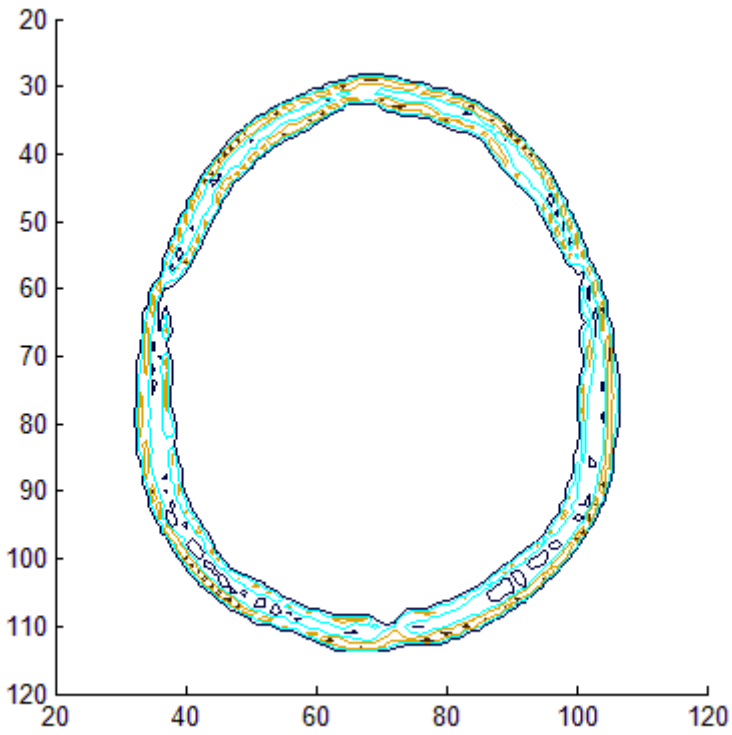
Šum su sive točke u cnom području koje se mogu vidjeti na slici 16. Kada bi se takve snimke provrtile kroz gore opisani program, podaci koje bismo dobili bili bi iznenađujuće loši. Na slijedećoj slici vidi se rezultat: nejasno definirana izopovršina.



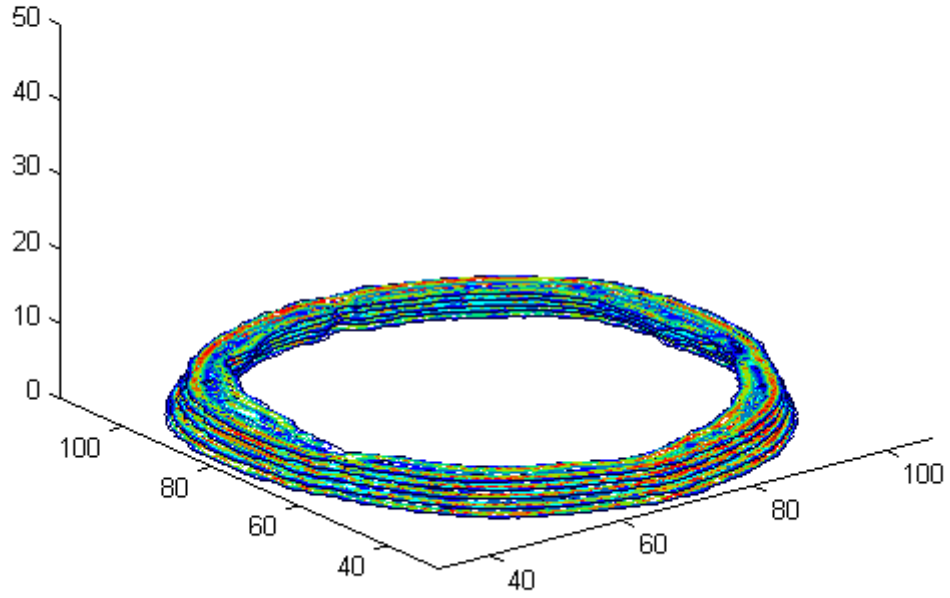
Slika 17. Nejasno definirana izopovršina uzrokovana pojavom šuma

Na gornjoj slici vidi se pokušaj stvaranja izopovršine, međutim podaci su toliko loši da Matlab nije znao gdje treba povezati izopovršinom, te je rezultat toga gornja slika. zaključuje se da je na slici previše podataka te se samanjuje na veličinu i kvalitetu slika sa prvog primjera. 128 x 128 piksela, 8-bitna.

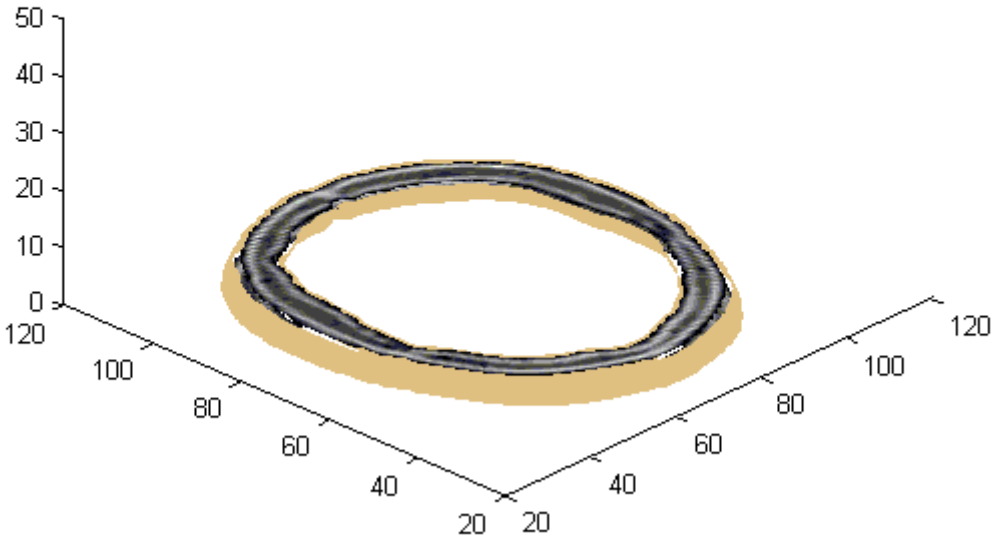
Čak ni tada, Matlab nije uspio izraditi odgovarajuću izopovršinu, već je trebalo „mrtve“ piksele izbaciti sa slike. To se radi na način da se svi pikseli vrlo slabog intenziteta jednostavno proglase potpuno crnim, tj. pridjeli im se vrijednost 255. Postupak se ponovi na svim slikama te se u konačnici stvore zadovoljavajući rezultati. Na slijedećim slikama, prikazan je postupak za 5 presjeka koji su bili podvrgnuti filtriranju i smanjenju rezolucije:



Slika 18. Definiranje rubnih kontura nakon uklonjenog šuma



Slika 19. Slaganje presjeka sa izraženim rubnim konturama bez šuma



Slika 20. Povezivanje rubnih kontura izopovršinom nakon uklonjenog šuma

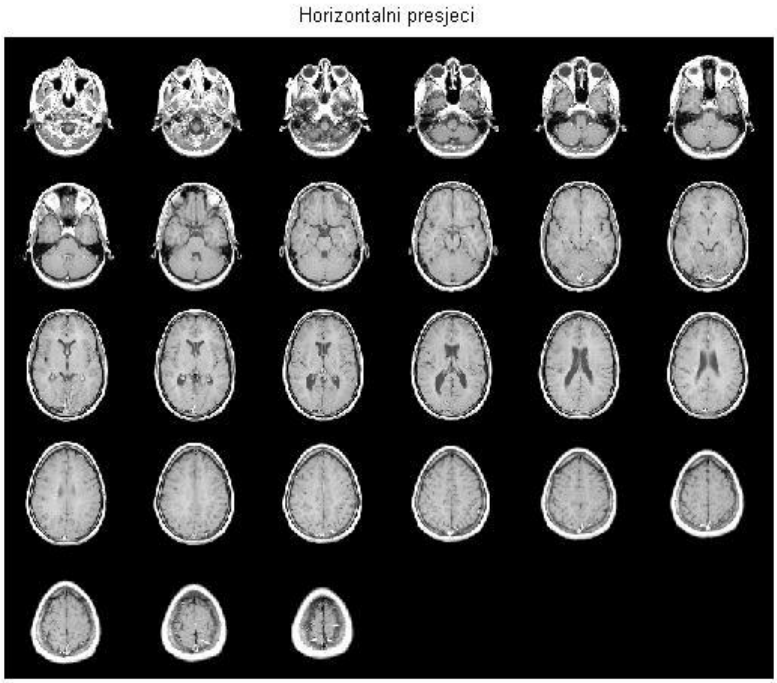
Iz gornjih slika vidi se da čim su „mrtvi“ pikseli izbačeni i čim su konture presjeka jasno izražene realni prikaz je jasan, i Matlabu nije problem prepoznati rubove. Ovaj postupak ponoviti će se za cijeli set originalnih MR snimaka, i to ne sa izraženom samo lubanjom već cijelom strukturom glave i mozga. Takav set snimaka može jasno izraziti te iskušati sve alate koje program „Brainiac“ nudi.

6.3. Prikaz sagitalnih i koronarnih presjeka

Jednom kada je 3D prikaz spremljen u obliku 3D matrice, lako je dobiti i presjeke u ostale dvije ravnine. Dobivanje sagitalnog i koronalnog presjeka pokazat će se na primjeru Matlabovih 27 snimaka.

```
load mri;
figure;
immovie(D,map); %stvara film iz višeokvirnog prikaza
montage(D,map); %ispisuje više prikaza kao pravokutna montaža
title('Horizontalni presjeci'); %dodaje tekst na vrh trenutne osi
```

Nakon prikazanog koda prikazuje se slijedeća slika:



Slika 21. Pregled aksijalnih presjeka na jednoj slici

Na gornjoj slici vidi se montaža svih presjeka iz martice „mri“ na jednom mjestu. Ista montaža koristiti će se za prikaz presjeka u ostale dvije ravnine. Uzimajući drugačije vrijeme uzorkovanja i drugačiju orijentaciju matrice stvara se sagitalni presjek:

```
M1 = D(:,64,:,:); size(M1) %izvlači sve potrebne podatke za okomiti presjek
```

Još uvijek se ne može vidjeti slika jer je to 127*1*1*27 matrica. Koristi se reshape (ili squeeze) da se dobije 127*27 te nakon toga se može sa (imshow) prikazati slika:

```
M2 = reshape(M1,[128 27]); size(M2)
figure, imshow(M2,map);
title('Sagitalni - Sirov podatak'); % dodaje naslov iznad slike
```

Nakon što se izvrši gore prikazani kod, u zasebnom prozoru ispisuje se slijedeća slika:

Slika 22. Izvlačenje sagitalnog presjeka iz skupa aksijalnih- sirova informacija

Gornja slika je nejasna te predstavlja „sirov“ podatak izvučen iz vodoravnih presjeka. Potrebno je prikaz modificirati kako slijedi:

```
T0 = maketform('affine',[0 -2.5; 1 0; 0 0]); % mjenja orijentaciju slike te
                                             % povećava frekvenciju
                                             % uzorkovanja za 2.5

R2 = makeresampler({'cubic','nearest'},'fill'); %stvara strukturu sa
                                                %drukčijim uzorkovanjem
M3 = imtransform(M2,T0,R2); %transformira sliku M2 koristeći T0 parametre
```

```
figure, imshow(M3,map);      %prikaz slike
title('Sagitalni presjek')   %naslov iznad slike
```

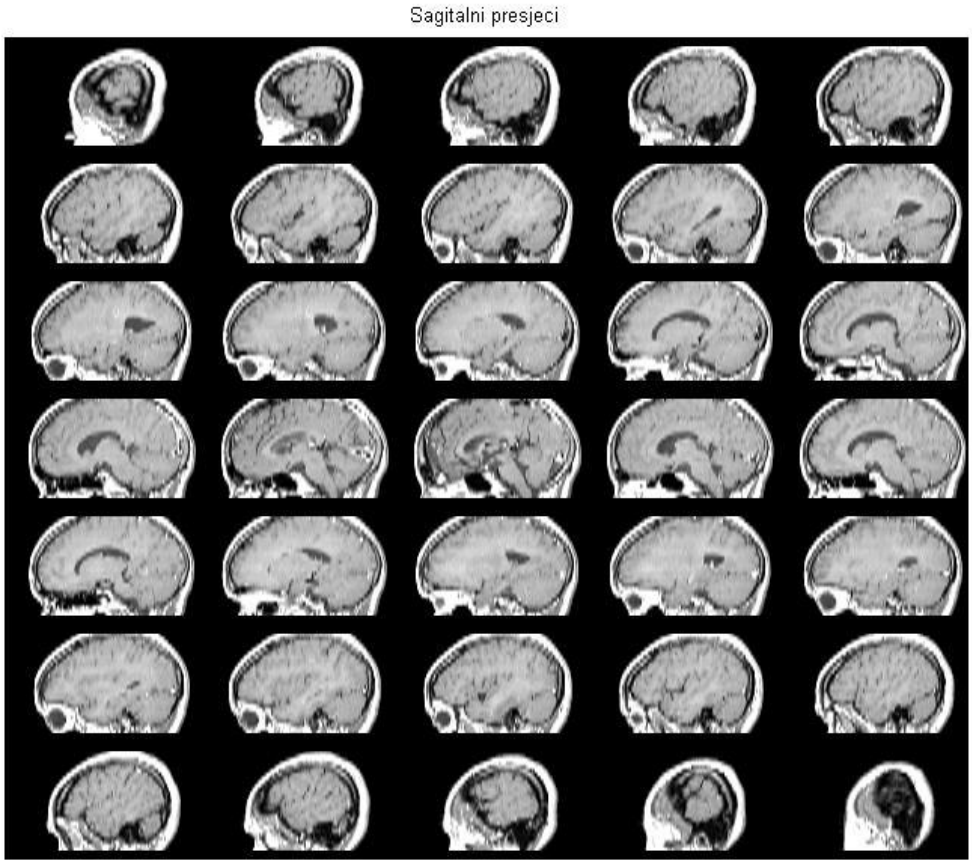
Te se pokretanjem gornjeg koda prikazuje slijedeće:

Slika 23. Sagitalni presjek, pravilno orijentiran i skaliran

Ukolko se želi prikazati čitav niz sagitalnih presjeka u jdnom pogldu to se postiže slijedećim kodom:

```
T1 = maketform('affine',[-2.5 0; 0 1; 68.5 0]); %skalira za -2.5 i stavlja
                                                 %pomak od 68.5
T2 = maketform('custom',3,2,[],@ipex003,64); %T2 i Tc mjenja 3D ulaz u
Tc = maketform('composite',T1,T2);           %2D izlaz

R3 = makeresampler({'cubic','nearest','nearest'},'fill');%stvara strukturu
sa drukčijim uzorkovanjem, ovaj put uzimajući treću dimenziju.


T3 = maketform('affine',[-2.5 0 0; 0 1 0; 0 0 0.5; 68.5 0 -14]);
%urđuje sliceve da budu posloženi

S = tformarray(D,T3,R3,[4 1 2],[1 2 4],[66 128 35],[],0);
%tformarray trensformira izlaz u 2D u jednom koraku
%uključuje 35 uzoraka od lijevo prema desno

figure;       % Uređivanje prikaza sliceva
immovie(S,map);
S2 = padarray(S,[6 0 0 0],0,'both');
montage(S2,map);
title('Sagitalni presjeci');
```

Generira se slijedeći prikaz.



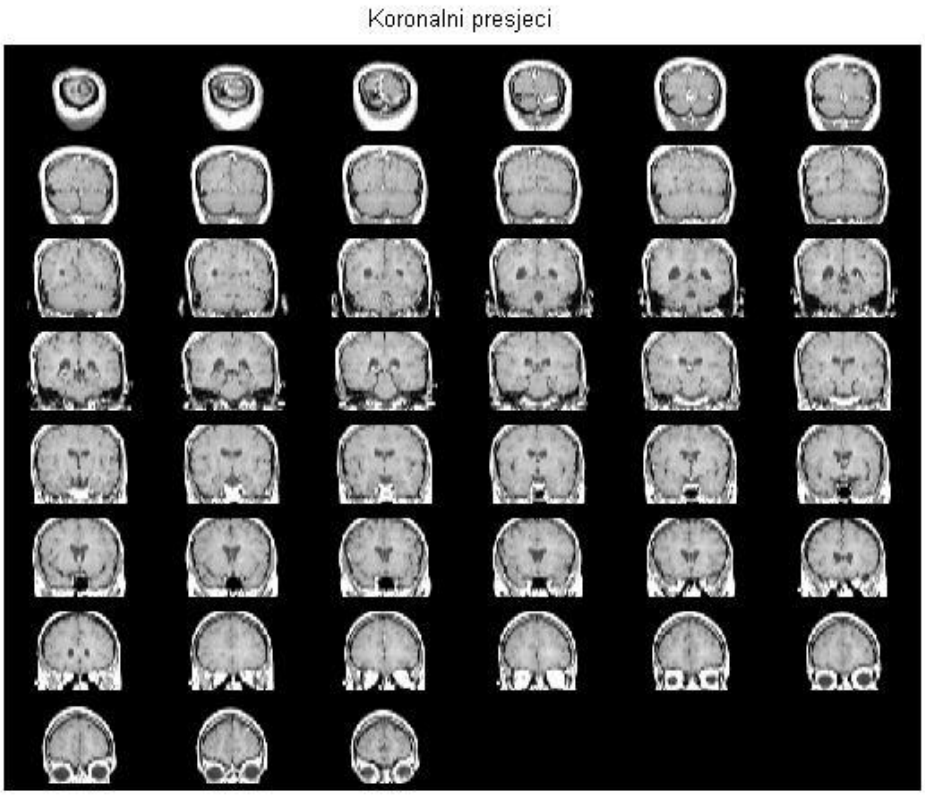
Slika 24. Prikaz sagitalnih presjeka na jednoj slici

Ovdje su prikazani svi sagitalni presjeci stvoreni od originalnih aksijalnih presjeka. Sagitalni presjeci su presjeci stvoreni usporedno s ravninom koja sijeće lubanju od jednog uha do drugog.

Za prikaz koronalnih presjeka te njihov prikaz na jednoj slici koristimo slijedeći kod:

```
T4 = maketform('affine',[-2.5 0 0; 0 1 0; 0 0 -0.5; 68.5 0 61]);
%Uređivanje sliceva
C = tformarray(D,T4,R3,[4 2 1],[1 2 4],[66 128 45],[],0);
%kao i u prethodnom koraku samo za ovaj presjek

figure;          %uređivanje prikaza sliceva
immovie(C,map); %slično kao i za sagitalne presjeke
C2 = padarray(C,[6 0 0 0],0,'both');
montage(C2,map);
title('Koronalni presjeci');
```

Nakon što se gornji kod izvrši u posebnom prozoru prikazuju se slijedeći presjeci.



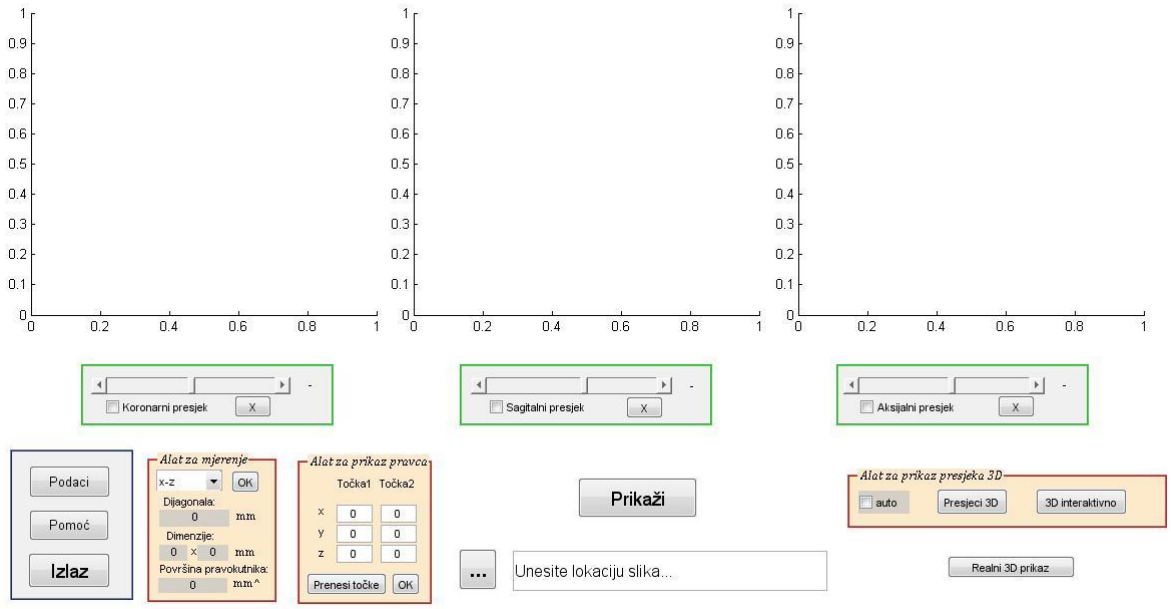
Slika 25. Izvlačenje koronarnih presjeka iz aksijalnih

Koronalni presjeci su presjeci stvoreni ravninom koja kreće od zatiljka prema čelu glave. Na taj način iz sloja aksijalnih presjeka dobiju se sagitalni i koronalni presjeci. Program najprije posloži aksijalne presjeke jedan na drugi da bi dobio volumen. Zatim mjenjajući uzorkovanje „sijeće“ stvoreni volumen u jednom, odnosno drugom smjeru. To je jedna od glavnih funkcija potrebnih za rad programa ovog rada.

7. Razvoj programa „Brainiac“

7.1. Opći izgled programa i funkcije

Program se razvijao u već spomenutom grafičkom korisničkom sućelju Matlaba. Sastoji se od glavnog prozora koji se rasteže preko cijelog ekrana, tako da stane što više podataka te da je prikaz presjeka jasniji, viš različitih tipki, koje služe za odabir podataka, klizaća, prozora za upis teksta, potvrdnih polja, i teksta. Početni izgled programa netom po pokretanju prikazan je na slici 26.



Slika 26. Osnovni izgled programa „Brainiac“

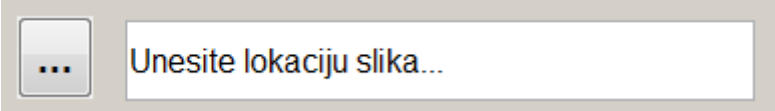
U gornjem dijelu programa prikazana su tri koordinatna sustava. Ovdje se na odabranim koordinatnim sustavima prikazuju presjeci: koronalni, sagitalni i aksijalni. Ispod koordinatnih sustava nalaze se klizači koji omogućuju odabir presjeka koji se prikazuje. U donjem lijevom kutu programa nalazi se prozorčić sa tri tipke. To su tipke opće namjene te služe za informacije i izlaz iz programa. Do njega je slijedeći prozorčić koji služi za mjerenje objekata na presjecima. Do njega je alat za prikaz pravca. U donjem središnjem dijelu programa je traka za upis (ili odabir) putanje gdje se nalazi datoteka koja sadržava Dicom snimke, te tipka „Prikaži“ koja prikazuje presjeke na osima.

U donjem desnom kutu je prozorčić sa tipkama za prikaz sva ti presjeka u jednom trodimenzionalnom koordinatnom sustavu. Na samom dnu lijevo je tipka koja prikazuje realan trodimenzionalan prikaz ljudske glave.

U nastavku proći će se kroz svaki dio programa posebno kako bi se detaljno objasnilo na koji način funkcionira pojedini dio programa. U ovom radu neće biti prikazan kod u cijelosti, nego pojedini dijelovi potrebni za rezumjevanje.

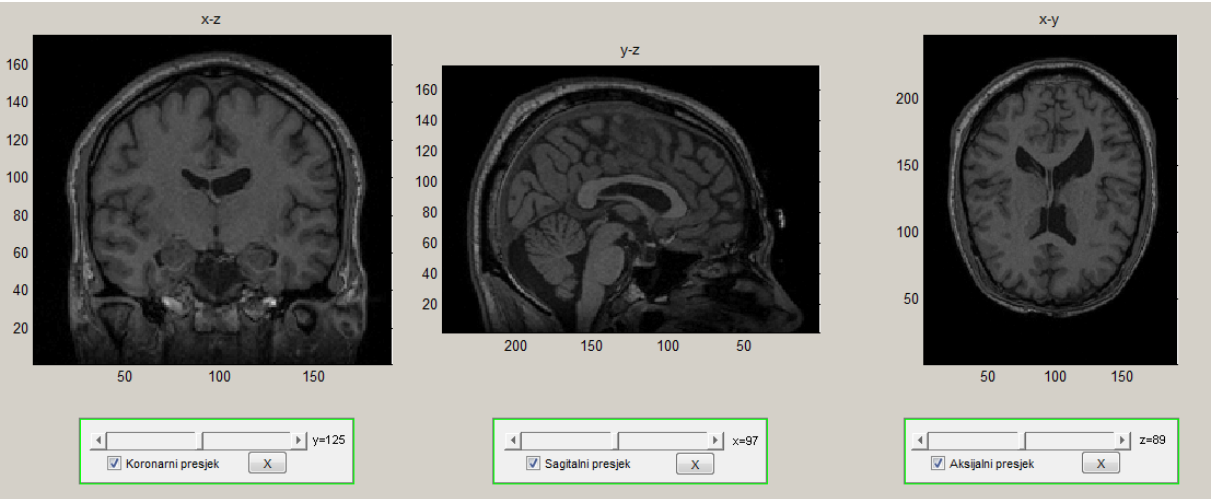
7.2. Prikaz dvodimenzionalnih presjeka

Najprije je potrebno unijeti putanju slika u odgovarajući prozor ili tražiti pritiskom na tipku „...“ kao na slici 27.



Slika 27. Polje za unos putanje direktorija sa DICOM snimkama

Zatim se tipkama za odabir, koje se nalaze ispod svakog koordinatnog sustava, odaberu presjeci koji se žele prikazati, te se pritiskom na tipku „Prikaži“ oni prikazju. Rezultat koji se dobije je prikazan na slijedećoj slici:



Slika 28. Prikaz koronarnog, sagitalnog i aksijalnog presjeka

U nastavku slijedi i objašnjenje koda koji je omogućio taj prikaz. Za otvaranje prozora za traženje datoteka služi slijedeći kod:

```
[filename, pathname] = uigetfile();%otvara prozor za traženje datoteka
if ~ischar(filename)
  return;
end
set(handles.edit5, 'String', pathname); %smješta putanju datoteka u polje
za unos
```

Kada se odabere neka slika, u prozorčić sa slike 27 upisuje se putanja gdje se nalazi ista. Za odabir nekog od prikaza mora se „čekirati“ kućica ispod željenog presjeka, kao što je prikazano na slici 29.

Slika 29. Odabir presjeka

Ako je u kučici kvačica tada se taj presjek prikazuje u pripadajućem koordinatnom sustavu. Pritiskom na tipku „Prikaži“ prikazuju se presjeci. Prije nego što se prikažu presjeci, program odradi slijedeći kod:

```
pathname = get(handles.edit5,'String'); %Uzima putanju datoteka
fileList = dir(fullfile(pathname, '*.dcm')); %Određuje tip datoteke
fileList = fileList(not([fileList.isdir])); %Stvara listu u koju će
spremati snimke
nFile = numel(fileList); %određuje broj snimaka
imgC = cell(1, nFile); %stvara ćeliju
for ii = 1:nFile; %for petlja stvara ćeliju onoliko veliku koliko ima slika
  imgC{ii} = dicomread(fullfile(pathname, fileList(ii).name));
end
imgSize = size(imgC{1}); %uzima rezoluciju slika i pstavlja kao dimenzije
ćelije
if all(cellfun('size', imgC, 1) == imgSize(1)) && ...
   all(cellfun('size', imgC, 2) == imgSize(2))
  img = cat(3, imgC{:});%stvaranje 3D matrice
else
  error('Slike su različite rezolucije.'); %filter za slike različite
rezolucije
end
img = squeeze (img); %odbacije nepotrebnu dimenziju matrice koja je
vrijednosti 1
assignin ('base','A',img) %sprema matricu u bazu podataka
a=img;
handles.vol=double(squeeze(a));
assignin('base','vol',handles.vol);
if ndims(handles.vol)~=3, %provjerava je li volumen 3D
    disp('Nije 3D')
    return
end
```

Ovaj dio koda izvlači slike iz prije odabranog direktorija te ih slaže jedan na drugi. Na taj način stvara volumen i sprema ga u glavni prostor za rad pod naziv „vol“. Treba spomenuti da broj i naziv odabranih slika može biti različit. Ukoliko je rezolucija različita među slikama program vraća upozorenje: „Slike su različite rezlucije“, a ukoliko stvaranje volumena nije uspijelo prikazuje se poruka: „Nije 3D“. Kod koji odlučuje hoće li se prikazati presjek, ovisno o tome je li kućica za prikaz označena ili nije, je obična if petlja a glasi:

```
[handles.sx,handles.sy,handles.sz]=size(handles.vol);
if get(handles.checkbox12,'value') %ako je kvačica uključena postavlja se
    set(handles.slider1,'min',1); %maksimalna i minimalna vrijednost broja
    set(handles.slider1,'max',handles.sx); %presjeka.
    set(handles.slider1,'SliderStep',[1/handles.sx , 10/handles.sx]);
%korak klizača
    set(handles.slider1,'value',round(handles.sx/2)+1); %postavljanje u
središnji početni položaj
else
    axes(handles.axes1); %ucrtava sliku sa smješkom ukoliko nije ucrtan
    im=imread('smjesko.png'); %presjek
    image(im);
end
```

Prva linija gornjeg koda odvaja maksimalan broj piksela u x, y i z smjeru. Ukoliko kućica za prikaz nije označena, umjesto presjeka u koordinatni sustav prikazati će se slika sa žutim smješkom. Prva i glavna funkcija programa je izvršena. Sada se, različitim alatima dostupnim u sklopu programa mogu analizirati presjeci. Ukoliko se želi prikazati neki drugi od presjeka u nizu, pomoću klizača, direktno ili pomoću strelica klizača na rubovima, namjesti se željena vrijednost te se prikazuje odabrani presjek. Pored klizaca je broj koji označuje koji je presjek u nizu prikazan. Da bi se gornje akcije omogućile potrebno je u programu imati slijedeći programski kod:

```
handles.vol=evalin('base','vol'); %uzima ranije spremljeni volumen
s1=round(get(handles.slider1,'value')); %uzima vrijednost sa klizača
s2=round(get(handles.slider2,'value')); %uzima vrijednost sa klizača
s3=round(get(handles.slider3,'value')); %uzima vrijednost sa klizača
assignin('base','s3',s3); %zapisuje vrijednost klizača u bazu podataka
assignin('base','s2',s2); %zapisuje vrijednost klizača u bazu podataka
assignin('base','s1',s1); %zapisuje vrijednost klizača u bazu podataka

vmx=max(handles.vol(:));%maksimalna vrijednost klizača
vmn=min(handles.vol(:));%minimalna vrijednost klizača

if get(handles.checkbox12,'value') %provjera je li radnja označena
    s1=round(get(handles.slider1,'value')); %uzima vrijednost klizača
if any(n==1) % x -- y z
set(handles.text1,'string',['y=',num2str(s1)]) %ispisuje trenutnu
vrijednost klizača
```

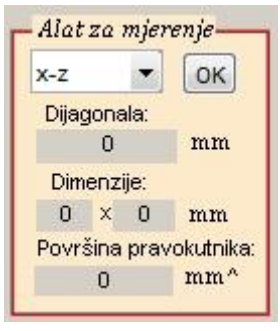
```
handles.vol=evalin('base','vol'); %uzimanje podataka iz baze podataka
s1=evalin('base','s2'); %uzimanje podataka iz baze podataka
s2=evalin('base','s1'); %uzimanje podataka iz baze podataka
s3=evalin('base','s3'); %uzimanje podataka iz baze podataka


colormap(gray) % mapa koja sadrži paletu boja koja se koristi
axes(handles.axes1); %odabir aktivnih osi
hslc=slice(handles.vol,[],s2,[]); %postavlja presjek na os
daspect([1,1,1]); %omjer prikaza podataka
axis tight; set(hslc(1),'LineStyle','none'); %dodatne opcije osi
az = 0;
el = 0;
view(az, el); %postavljanje pogleda
     title('x-z') %postavljanje naslova iznad osi

end
end

if get(handles.checkbox13,'value') %isto kao gornji postupak samo za
slijedeći koordinatni sustav
    s2=round(get(handles.slider2,'value'));
if any(n==2)% y -- x z
     set(handles.text2,'string',['x=',num2str(s2)])
     handles.vol=evalin('base','vol');
s1=evalin('base','s2');
s2=evalin('base','s1');
s3=evalin('base','s3');
colormap(gray)
axes(handles.axes2);
hslc=slice(handles.vol,s1,[],[]);
daspect([1,1,1]);
axis tight; set(hslc(1),'LineStyle','none');
az = -90;
el = 0;
view(az, el);
     title('y-z')
end
end

if get(handles.checkbox14,'value') %isto kao gornji postupak samo za
slijedeći koordinatni sustav
    s3=round(get(handles.slider3,'value'));%#ok
if any(n==3)% z -- x y


handles.vol=evalin('base','vol');
s1=evalin('base','s2');%#ok
s2=evalin('base','s1');%#ok
s3=evalin('base','s3');

colormap(gray)
axes(handles.axes3); %#ok

hslc=slice(handles.vol,[],[],s3);%rotate3d on;
daspect([1,1,1]);
axis tight; set(hslc(1),'LineStyle','none');
az = 0;
el = 90;
```

```
view(az, el);

    set(handles.text3,'string',['z=',num2str(s3)])
    title('x-y')
end
end
```

Prikazani kod obavlja više stvari. Prvi dio prikuplja podatke od klizača i polja za odabir pojedinog presjeka drugi dio postavlja nove vrijednosti klizača, a treći prikazuje presjeke sa pripadajućim nazivima osi.

7.3. Alat za mjerenje

Alatom za mjerenje može se na presjeku izmjeriti udaljenost dvije odabrane točke, odrediti dimenzije pravokutnika kojeg te dvije točke omeđuju, te odrediti površina nekog željenog pravokutnog područja. Alat je koristan prilikom mjerenja neželjene pojave unutar strukture mozga, ili ako se jednostavno želi izmjeriti udaljenost dviju područja unutar mozga. Područje koje ispisuje podatke sastoji se od tri dijela. Prvi dio alata za mjerenje, nakon što su se točke na pravilan način odabrale, automatski računa i prikazuje udaljenost između tih točaka, tj. računa dijagonalu kvadrata omeđenog točkama. Drugi dio prikazuje dimenzije kvadrata, tj duljinu a i b stranice pravokutnika. Treći dio automatski računa površinu i ispisuje je unutar alata za mjerenje. Cijeli alat za mjerenje u programu izgleda kao na slici 30.



Slika 30. Alat za mjerenje

Da bi se nabrojane funkcije obavile najprije treba u padajućem meniju odabrati koordinatne osi na kojima želimo mjeriti. Kod koji omogućuje odabir osi glasi:

```
graf=get(handles.popupmenu2,'value'); %sklopka koja odabire aktivni
koordinatni sustav ovisno o prije odabranoj opciji
switch graf
    case 1
        graf='axes1';
    case 2
        graf='axes2';
    case 3
        graf='axes3';
end
assignin ('base','graf',graf); %zapisuje aktivni koordinatni sustav u bazu
podtaka
```

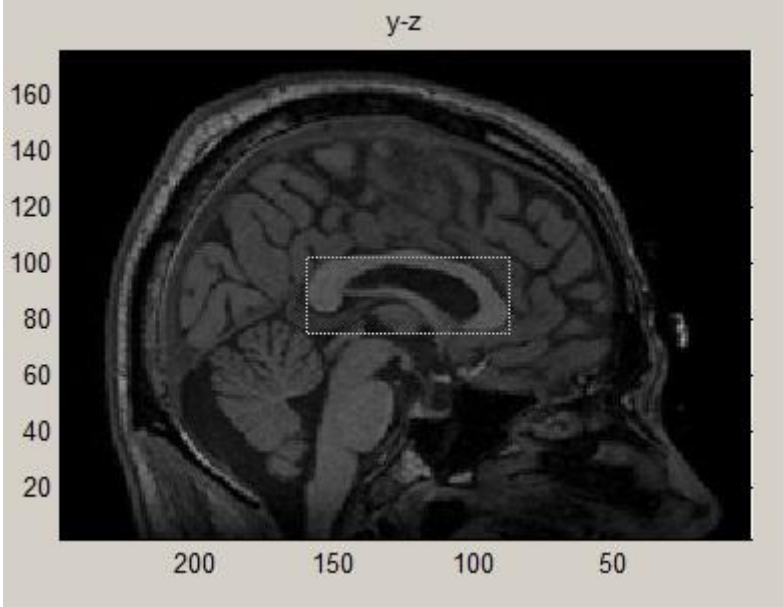
Prvi dio koda uzima vrijednost iz padajućeg menija te poput sklopke premješta izabrani koordinatni sustav u aktivni koordinatni sustav na kojem će se obaviti mjerenje. Drugi dio koda sprema izabrani koordinatni sustav u glavnu bazu podataka Matlaba za daljnje korištenje. Pritiskom na tipku „OK“, unutar alata, aktivira se mjerenje te je potrebno mišem označiti pravokutnik na presjeku. To se postiže tako da se klikom lijeve tipke miša označi jedan kraj te se ne puštajući tipku miša odvuče miš na drugi kraj kvadrata. Za pomoć u predočavanju kvadrata stvara se kvadrat opcrtan točkastom linijom. To se postiže slijedećim linijama koda:

```
graf = evalin('base','graf'); %postavljanje aktivnog grafa
axes(handles.(graf));
k=waitforbuttonpress; %čekanje na pritisak tipke miša
point1 = get(gca,'CurrentPoint'); %uzimanje koordinata prve točke
assignin('base','point1',point1); %spremanje koordinata prve točke
finalRect = rbbox;  %prikaz točkastog kvadrata do druge točke
point2 = get(gca,'CurrentPoint'); %uzimanje koordinata druge točke
assignin('base','point2',point2); %spremanje koordinata druge točke
if graf == 'axes1'%provjeravanje koja je os aktivna
    c = sqrt(((point1(1)-point2(1))^2)+((point1(5)-point2(5))^2)); %računa
udaljenost jede točke do druge pitagorinim poučkom
    assignin ('base','c',c); %sprema vrijednost
    set(handles.text9,'String',c); %ispisuje vrijednost
    set(handles.text21,'String',abs(point1(1)-point2(1))); %računa i
ispisuje duljinu stranice pravokutnika
    set(handles.text23,'String',abs(point1(5)-point2(5))); %računa i
ispisuje duljinu stranice pravokutnika
    set(handles.text26,'String',abs(point1(5)-point2(5))*abs(point1(1)-
point2(1))); %ispisuje površinu pravokutnika
elseif graf == 'axes2'%isti postupak kao gore samo za drugu os, ukoliko je
ona aktivna
    c = sqrt(((point1(3)-point2(3))^2)+((point1(5)-point2(5))^2));
    assignin ('base','c',c);
    set(handles.text9,'String',c);
    set(handles.text21,'String',abs(point1(3)-point2(3)));
```

```
    set(handles.text23,'String',abs(point1(5)-point2(5)));
    set(handles.text26,'String',abs(point1(5)-point2(5))*abs(point1(3)-
point2(3)));
else %isti postupak kao gore samo za treću os, ukoliko je ona aktivna
    c = sqrt(((point1(1)-point2(1))^2)+((point1(3)-point2(3))^2));
    assignin ('base','c',c);
    set(handles.text9,'String',c);
    set(handles.text21,'String',abs(point1(1)-point2(1)));
    set(handles.text23,'String',abs(point1(3)-point2(3)));
    set(handles.text26,'String',abs(point1(3)-point2(3))*abs(point1(1)-
point2(1)));
end
```

Prvi dio koda, za aktivni koordinatni sustav iz glavne baze podataka, uzima već od prije spremljen koordinatni sustav. Zatim čeka na pritisak tipke miša, nakon čega sprema podatke o točci na kojoj se pokazivač u tom trenutku našao. Nakon što se pokazivač odvuće na drugo mjesto te se tipka pusti, program sprema podatke o drugoj točci. Podaci koji se spremaju su x i y koordinate točaka.

Drugi dio koda preko trigonometrije računa dijagonalu, dimenzije te površinu opcrtanog pravokutnika. Primjer kako alat radi prikazan je na slici 31.



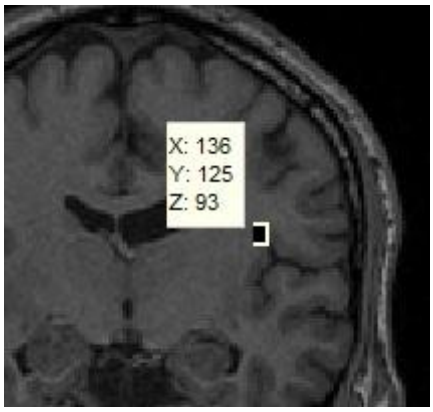
Slika 31. Prikaz rada alata za mjerenje

7.4. Alat za prikaz pravca

Alat za prikaz pravca je jedna od glavnih funkcija programa, te je uvijet za prikaz točaka prodora na pojedinim presjecima. Alat radi na principu da uzima koordinate dviju točaka, te pomoću jednadžbe pravca crta pravac u trokoordinatnom pregledu svih triju presjeka. Točke mogu biti uzete sa pojedinih dvodimenzionalnih prikaza, trodimenzionalnog prikaza svih presjeka ili realnog trodimenzionalnog prikaza. O pregedu sva tri presjeka i realnom 3D prikazu biti će riječi kasnije. Alat izgleda kao na slici 32.

Slika 32. Alat za prikaz putanje alata

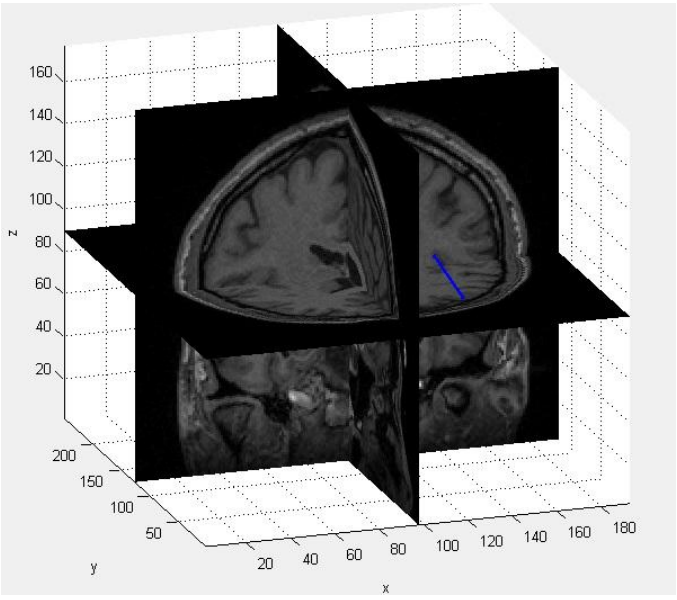
Alat se sastoji od 6 polja gdje se mogu koordinate točaka upisati ručno ili prenijeti sa točaka koje smo odabrali pokazivačem na nekom od presjeka. Ukoliko su točke odabrane sa presjeka, nakon odabira, pritiskom na „Prenesi točke“ koordinate točaka se upisuju u polja. Točke se sa presjeka uzimaju tako da se, najprije, sa alatne trake odabere „Data cursor“, a zatim se klikne na neko područje na presjeku. Na mjestu gdje je točka odabrana pojaviti će se koordinate u x, y i z smjeru. Pritiskom desne tipke miša na mjesto odabira točke otvara se padajući meni gdje treba pritisnuti „Export data cursor to workspace“ (*engl.* Prenesi podatke u glavnu bazu podataka). Jednu točku potrebno je nazvati „T1“ a drugu „T2“. Nakon što se postupak ponovi i za drugu točku, pritiskom na tipku „Prenesi točke“ u polja se upisuju koordinate točaka. Kako taj postupak izgleda prikazano je na slijedećm slikama.



Slika 33. Odabir koordinata točaka sa presjeka

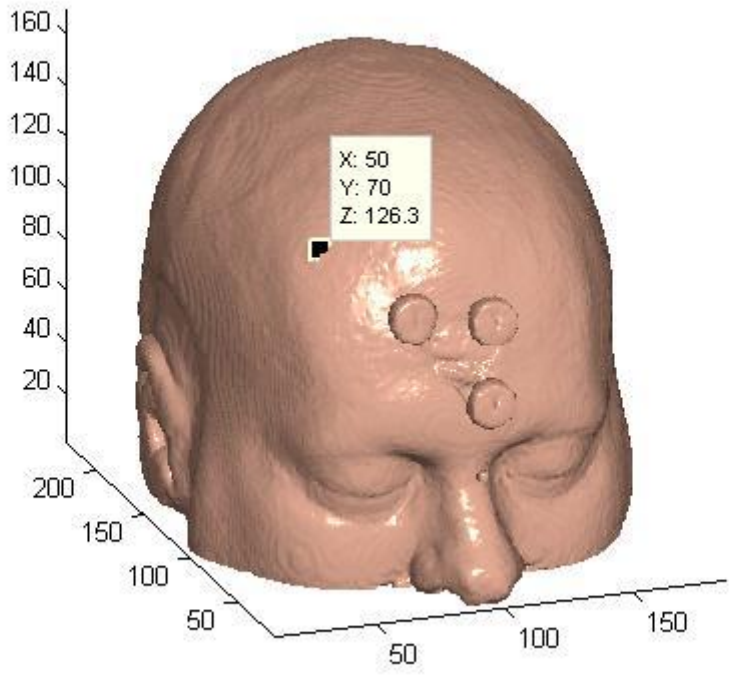
Slika 34. Prijenos odabranih točaka u bazu podataka

Nakon što su koordinate točaka prenesene pritiskom na tipku „OK“ u trokoordinatnom sustavu svih presjeka ucrtava se pravac koji spaja te dvije točke te predstavlja putanju kojom će operacijski alat probijati stvarno tkivo. Kako ucrtani pravac izgleda prikazano je na slici 35.



Slika 35. Prikaz ucrtane putanje pravca

Točke mogu biti odabrane i sa realnog presjeka kao što je prikazano na slici 36.



Slika 36. Primjer odabira koordinata sa realnog presjeka

„Data cursor“ je implementirani alat Matlaba, te neće biti pomnije objašnjen. Programski kod koji prenosi uzete točke glasi:

```
T1 = evalin('base','T1'); %uzima podatke točke 1
T2 = evalin('base','T2'); %uzima podatke točke 2
x1=T1.Position(1); %izdvaja poziciju točke 1 na osi x
x2=T2.Position(1); %izdvaja poziciju točke 2 na osi x
y1=T1.Position(2); %izdvaja poziciju točke 1 na osi y
y2=T2.Position(2); %izdvaja poziciju točke 2 na osi y
z1=T1.Position(3); %izdvaja poziciju točke 1 na osi z
z2=T2.Position(3); %izdvaja poziciju točke 2 na osi z

set(handles.edit7,'String',x1); %ispisuje vrijednost u programu
set(handles.edit8,'String',y1); %ispisuje vrijednost u programu
set(handles.edit9,'String',z1); %ispisuje vrijednost u programu
set(handles.edit13,'String',x2); %ispisuje vrijednost u programu
set(handles.edit14,'String',y2); %ispisuje vrijednost u programu
set(handles.edit15,'String',z2); %ispisuje vrijednost u programu
```

Prikazani programski kod jednostavno uzima podatke iz glavne baze podataka i smješta ih u polja odabrana polja u programu. Kod koji uzima podatke iz polja te ucrtava pravac između dvije odabrane točke glasi:

```
x1=str2double(get(handles.edit7,'String')); %vrijednost polja čini brojem
y1=str2double(get(handles.edit8,'String')); %vrijednost polja čini brojem
z1=str2double(get(handles.edit9,'String')); %vrijednost polja čini brojem
x2=str2double(get(handles.edit13,'String')); %vrijednost polja čini brojem
y2=str2double(get(handles.edit14,'String')); %vrijednost polja čini brojem
z2=str2double(get(handles.edit15,'String')); %vrijednost polja čini brojem

assignin('base','x1kuc',x1); %zapisuje vrijednost kordinata u bazu podataka
assignin('base','x2kuc',x2); %zapisuje vrijednost kordinata u bazu podataka
assignin('base','y1kuc',y1); %zapisuje vrijednost kordinata u bazu podataka
assignin('base','y2kuc',y2); %zapisuje vrijednost kordinata u bazu podataka
assignin('base','z1kuc',z1); %zapisuje vrijednost kordinata u bazu podataka
assignin('base','z2kuc',z2); %zapisuje vrijednost kordinata u bazu podataka

line([x1 x2],[y1 y2],[z1 z2],'Marker','.','LineStyle','-','LineWidth',2);
%ucrtava pravac plave boje debljine 2 mm.
```

gdje prvi dio koda pretvara podatke upisane u polja iz tipkanog upisa u skalar,tako da program može računati sa tim podacima, a drugi jednostavnom funkcijom ucrtava plavi pravac debljine 2 mm kao što je prikazano na slici 35.

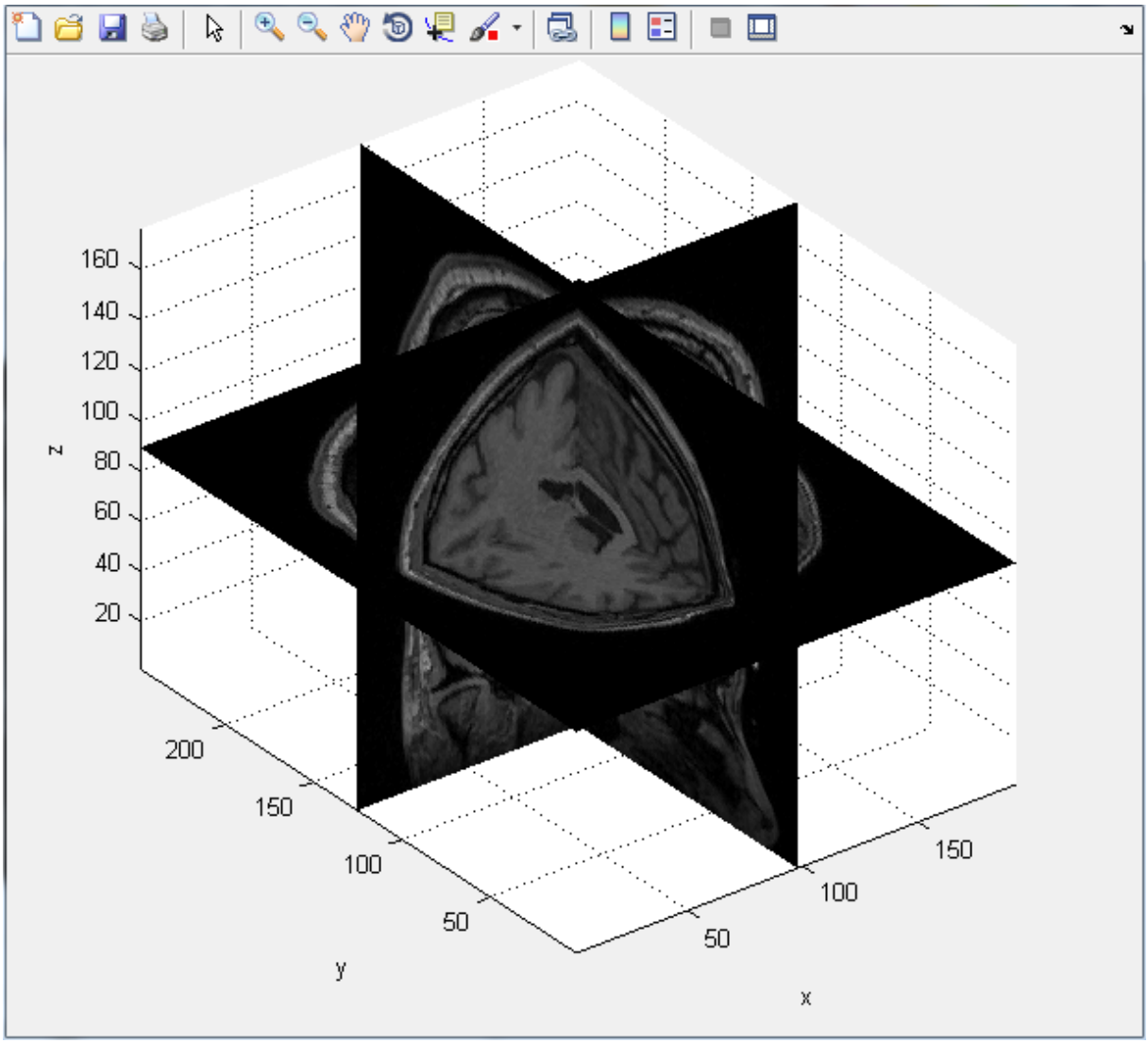
7.5. Prikaz svih presjeka u trodimenzionalnom koordinatnom sustavu

Jedna od posebitosti programa jest sposobnost da prikaže sve navedene presjeke: aksijalni, koronalni i sagitalni u jednom trodimenzionalnom koordinatnom sustavu radi boljeg shvaćanja i lakšeg predočavanja podataka. Nakon što su prikazani presjeci u pojedinim ravninama, mogu se prikazati i svi zajedno. Prikaz je vrlo koristan pri odabiru točaka putanje pravca ili za bolje vizualiziranje putanje pravca. Koristi se programski kod:

```
set(gcf,'toolbar','figure'); %postavlja alatnu traku
handles.vol=evalin('base','vol'); %uzima volumen iz baze podataka
s1=evalin('base','s2'); %uzima položaj drugug klizača
s2=evalin('base','s1'); %uzima položaj prvog klizača
s3=evalin('base','s3'); %uzima položaj trećeg klizača
```

```
colormap(gray) %postavlja mapu paleta boja
axes(handles.axes4); %postavlja aktivnu os
hslc=slice(handles.vol,s1,s2,s3); %sijeće volumen u tri ravnine i prikazuje
u aktivnoj osi
axis tight; set(hslc(1:3),'LineStyle','none'); %dodatni parametri osi
xlabel 'x' ;ylabel 'y' ;zlabel 'z'; %postavljanj oznaka na x, y i z os.
daspect([1,1,1,]); %omjer prikaza podataka
```

Prikazani programski kod najprije postavlja osnovnu alatnu traku na prozor gdje će se prikazati presjeci. Zatim iz glavne baze podataka uzima stvoreni volumen „vol“. Funkcijom „slice“ sijeće volumen te presjeke ispisuje u jednom prozoru. Funkcija „daspect“ postavlja omjer podataka kako bi osi koordinatnih sustava bile proporcionalne. Pritiskom na tipku „Presjeci 3D“ otvara se prozor kao na slici 37.



Slika 37. Prikaz sagitalnog, koronarnog i aksijalnog presjeka

U prozoru se stvara trodimenzionalni koordinatni sustav sa ucrtanim aksijalnim, sagitalnim i koronalnim presjekom. Prikazuju se presjeci koji su, ranije, bili odabrani pomoću klizača. Sustav se može rotirati pomoću alata „Rotate 3D koji se nalazi na alatnoj traci prikazanog prozora.

Prikaz svih presjeka na jednom jestu može se i ostvariti u interaktivnoj verziji gdje je moguće tipkom miša interaktivno pomicati presjeke uzduž okomite osi. Na taj način još je više olakšano manipularanje presjecima, te opće shvaćanje problema i planiranje zahvata buduće operacije. Dovoljno je mišem prijeći preko presjeka koji se želi pomicati te se pritiskom lijeve tipke miša presjek odvuće na željeno mjesto. Postupak se može ponoviti nebrojeno puta za presjeke u sve tri ravnine. Presjek je isti kao i što je prikazano na slici 37 s time da ovaj put korisnik interaktivno sudjeluje. Programski kod koji ostvaruje gore opisanu funkciju glasi:

```
pathname = get(handles.edit5,'String'); %Uzima putanju datoteka
fileList = dir(fullfile(pathname, '*.dcm')); %Određuje tip datoteke
fileList = fileList(not([fileList.isdir])); %Stvara listu u koju će
spremati snimke
nFile = numel(fileList); %određuje broj snimaka
imgC = cell(1, nFile); %stvara ćeliju
for ii = 1:nFile; %for petlja stvara ćeliju onoliko veliku koliko ima slika
  imgC{ii} = dicomread(fullfile(pathname, fileList(ii).name));
end
imgSize = size(imgC{1}); %uzima rezoluciju slika i pstavlja kao dimenzije
ćelije
if all(cellfun('size', imgC, 1) == imgSize(1)) && ...
   all(cellfun('size', imgC, 2) == imgSize(2))
  img = cat(3, imgC{:});%stvara 3D volumen ukoliko su snimke jednake
veličine
else
  error('Slike su drukčije veličine'); %ukoliko nisu iste veličine ispisuje
upozorenje
end

img=double(squeeze(img)); %uklanja nepotrebnu dimeziju matrice

figure('Name','Pregled presjeka 3D','NumberTitle','off','Colormap',gray);
%postavlja uvjete za otvaranje novog prozora naziva „Pregled presjeka 3D
[x,y,z] = meshgrid(1:size(V,2),1:size(V,1),1:size(V,3)); %postavljanje
mreže preko trokoordinatnog sustava proporcionalno velike podacima
daspect([1,1,1,]); %omjer prikaza podataka
hslices=slice(x,y,z,V,size(V,2)/2,size(V,1)/2,size(V,3)/2); %siječe volumen
u tri ravnine
colormap(gray); %mapa palete boja
xlabel('x');ylabel('y');zlabel('z'); %nazivi osi
xlim([1 size(V,1)*4]); %postavljanje limita osi
ylim([1 size(V,2)*4]); %postavljanje limita osi
zlim([1 size(V,3)*4]); %postavljanje limita osi
hXslice=hslices(1); %utvrđivanje koronalnog presjeka
hYslice=hslices(2); %utvrđivanje sagitalnog presjeka
hZslice=hslices(3); %utvrđivanje aksijalnog presjeka
```

```
set(hXslice,'EdgeColor','None','Tag','PresjekX'); %postavljanje presjeka
set(hYslice,'EdgeColor','None','Tag','PresjekY'); %postavljanje presjeka
set(hZslice,'EdgeColor','None','Tag','PresjekZ'); %postavljanje presjeka

pomakX(hXslice,V); %funkcija koja omogućuje interaktivni pomak osi
pomakY(hYslice,V); %funkcija koja omogućuje interaktivni pomak osi
pomakZ(hZslice,V); %funkcija koja omogućuje interaktivni pomak osi

axis tight;
axis vis3d; %dodatne opcije koordinatnog sustava
set(gca,'ZDir','normal'); %postavljanje orijentacije podataka
set(gca,'YDir','normal');
daspect([1,1,1,]); %postavljanje omjera podataka
```

Prvi dio programskog koda izvlači presjeke iz datoteke koja je upisana u prozoru za upis putanje, pretvara ih u volumen te slaže presjeke po osima. Postavlja ih tako da su svi presjeci na početku u središnjoj poziciji. Drugi dio koda je funkcija „pomak“ koje se poziva iz glavne baze podataka, a kada je aktivirana postiže mogućnost interaktivnog pomicanja presjeka. Slijedi primjer koda „pomak“ smjeru x:

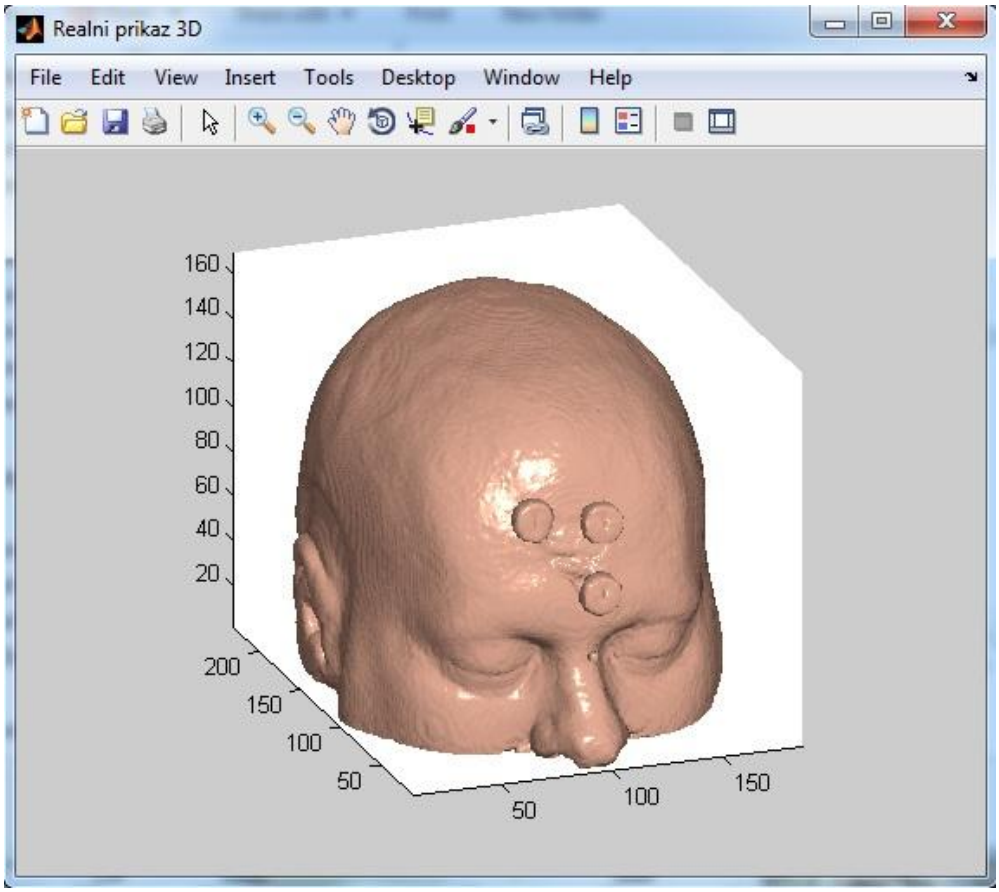
```
function pomakX(h,A)

gui = get(gcf,'UserData'); %uzimanje početnog položaja
set(h,'ButtonDownFcn',@startmovit); %praćenje pomaka miša
set(gcf,'UserData',{gui;A});%pamćenje završnog položaja

function startmovit(src,evnt)
temp = get(gcf,'UserData'); %izvlačenje podataka iz trenutno aktivnog
gui=temp{1};%koordinatnog sustava
A=temp{2};

set(gcf,'PointerShapeCData',nan(16,16)); %postavlanje oblika pokazivača
set(gcf,'Pointer','custom'); %oslobađanje vođenja pokazivača miša

gui.currenthandle = src; %uzimanje trenutnog sidra pokazivača
thisfig = gcbf();%uzimanje trenutnog sidra aktivnog prozora
set(thisfig,'WindowButtonMotionFcn',@movit); %prikaz pomaka presjeka
set(thisfig,'WindowButtonUpFcn',@stopmovit); %zaustavlja pomak presjeka

gui.startpoint = get(gca,'CurrentPoint'); %početna točka
set(gui.currenthandle,'UserData',{get(gui.currenthandle,'XData')
%postavljanje trenutnih podataka o položaju
get(gui.currenthandle,'YData') %postavljanje trenutnih podataka o položaju
get(gui.currenthandle,'ZData')}); %postavljanje trenutnih podataka o
položaju
set(gcf,'UserData',{gui;A});%postavljanje trenutnih podataka o položaju

function pomak(src,evnt)
temp = get(gcf,'UserData'); %izvlačenje podataka
gui=temp{1};%izvlačenje podataka iz trenutno aktivnog koordinatnog
A=temp{2};%sustava
```

```
try
if isequal(gui.startpoint,[]) %ukoliko su početna i krajnja točka iste ne
poduzima se nikakva akcija
    return
end
catch
end

XYData = get(gui.currenthandle,'UserData'); %izvlačenje podataka
pos = get(gca,'CurrentPoint')-gui.startpoint; %izvlačenje podataka

set(gui.currenthandle,'XData',XYData{1} + pos(1,1)); %postavljanje novih
aktivnih podataka
XData=XYData{1};
XData=max(XData(:));
XData=round(XData + pos(1,1)); %ukoliko je pokazivač miša stao na nekom
decimalnom broju, presjek ne postoji, tako da se treba vrijednost
zaokružiti na cijeli broj kako bi se prikazao presjek
if XData > size(A,2) %omogućuje da presjeci ne mogu pobijeći iz prozora
nego se zaustavlja na zadnjem presjeku
    XData=size(A,2);
end
ImgCData=squeeze(A(:,XData,:)); %uklanjanje nepotrebne dimenzije 3D matrice
set(gui.currenthandle,'CData',ImgCData); %postavljanje novog sidra
slicehandle=get(gui.currenthandle);
drawnow;
set(gcf,'UserData',{gui;A});%postavljanje novih informacija u bazu podataka

function stopmovit(src,evnt) %funkcija za prestanak gibanja

thisfig = gcbf();%uzimanje sidra trenutnog prikaza

temp = get(gcf,'UserData'); % izvlačenje podataka iz trenutno aktivnog
koordinatnog sustava
gui=temp{1};
A=temp{2};

set(gcf,'Pointer','arrow'); %postavljanje pokazivača u obliku strelice
set(thisfig,'WindowButtonUpFcn',''); %aktivira se na puštanje tipke miša
set(thisfig,'WindowButtonMotionFcn',''); %privid pomaka pokazivača
drawnow;
set(gui.currenthandle,'UserData',''); %postavlanje novih podataka
set(gcf,'UserData',{gui;A});%spremanje novih informacija u bazu podataka
```

Prvi dio navedenog programskog koda postavlja uvjete za početak izvršenja funkcije, tako da se radnja aktivira tek pritiskom lijeve tipke miša. Drugi dio koda prati pomak miša, računa njegove koordinate, te obnavlja položaj presjeka, brišući posljednji tako da se dobije privid kretanja presjeka uzduž osi. Treći dio koda brine se o tome da, kada se pusti tipka miša, program stane sa osvježavanjem podataka tako da zadnji presjek ostane prikazan. Opcija interaktivnog pomicanja presjeka korisna je prilikom planiranja putanje medicinskog alata radi lakšeg razumijevanja problema.

7.6. Realni 3D prikaz

Prilikom shvaćanja nekih podataka ljudi imaju običaj povezivati ih sa pojavama u stvarnom svijetu kako bi prije shvatili neku pojavu. Vrlo je teško na osnovu niza horizontalnih snimaka stvoriti predođbu o tome kako bi ta ljudska glava izgledala u stvarnosti. Ukoliko bi to programom bilo omogućeno, tada bi shvaćanje problema, te riješavanje istog bilo uvelike olakšano. Upravo iz tog razloga, u program je ugrađen alat koji slaže presjeke jedan na drugi, te ih povezujući izopovršinom boje ljudske kože, stvara prilično realan prikaz ljudske glave u stvarnosti. Realni prikaz moguće je dobiti ukoliko je u polje za upis putanje upisana putanja datoteke koja sadrži Dicom snimke, te ukoliko se pritisne tipka „Realni 3D prikaz“. Prikazuje se prozor kao na slici 38.



Slika 38. Realni trodimenzionalni prikaz

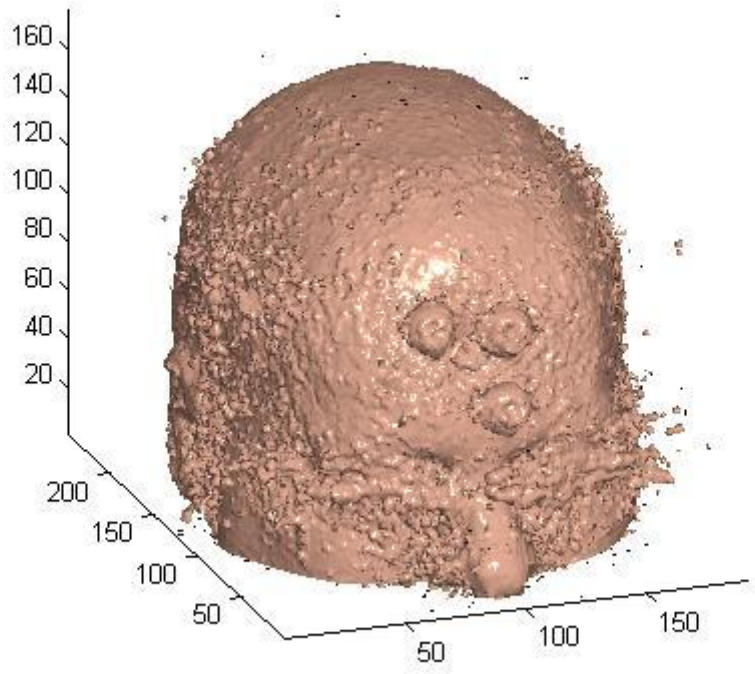
Prikaz je moguće rotirati kako bi se dobila što bolja percepcija pacijentove strukture glave. Ovaj prikaz vrlo je koristan kada se treba odabrati početna točka prodora

medicinskog alata prilikom operacije. Programski kod koji se koristi za realni prikaz je slijedeći:

```
pathname = get(handles.edit5,'String'); %Uzima putanju datoteka
fileList = dir(fullfile(pathname, '*.dcm')); %Određuje tip datoteke
fileList = fileList(not([fileList.isdir])); %Stvara listu u koju će
spremati snimke
nFile = numel(fileList); %određuje broj snimaka
imgC = cell(1, nFile); %stvara ćeliju
for ii = 1:nFile; %for petlja stvara ćeliju onoliko veliku koliko ima slika
  imgC{ii} = dicomread(fullfile(pathname, fileList(ii).name));
end
imgSize = size(imgC{1}); %uzima rezoluciju slika i pstavlja kao dimenzije
ćelije
if all(cellfun('size', imgC, 1) == imgSize(1)) && ...
   all(cellfun('size', imgC, 2) == imgSize(2))
  img = cat(3, imgC{:});%stvaranje 3D matrice
else
  error('Slike su različite rezolucije.'); %filter za slike različite
rezolucije

        for i = 1 : size(img,1) %for petlja koja izbacije šum tako da
pretvara sve piksele jednakog ili slabijeg intenziteta od 135
            for j = 1 : size(img,2)
                for k = 1: size(img,3)
                    if img(i,j,k)<=135, img(i,j,k)=0;
                    end;
                end;
            end;
        end;


figure('Name','Realni prikaz 3D','NumberTitle','off','Colormap',gray)
img = squeeze (img); %postavljanje naslova, uklanjanje nepotrebne
Ds = smooth3(img);   %dimenzije i zaglađivanje slike

hiso = patch(isosurface(Ds,5),...
    'FaceColor',[1,.75,.65],...
    'EdgeColor','none');         %definiranje svojstava izpopvršine
isonormals(Ds,hiso)

hcap = patch(isocaps(img,5),...
    'FaceColor','interp',...
    'EdgeColor','none');         %definiranje svojstava izokapsi

view(-20,20) %postavljenj kuta gledanja

axis tight %postavljenje koordinatnih odi usko oko podataka
daspect([1,1,1]) %omjer prikaza podataka

lightangle(-10,30);  %postavljanje kuta rasvjete
set(gcf,'Renderer','zbuffer'); lighting phong%vrsta rasvjete
set(hcap,'AmbientStrength',.6)     %postavljanje ambijenta
set(hiso,'SpecularColorReflectance',0,'SpecularExponent',50) %uređivanje
pogleda da izgleda što realniji
```

Prvi dio programskog koda izvlači Dicom snimke iz direktorija, slaže ih jedan na drugi te stvara početni trodimenzionalni volumen. Kada bi se taj volumen ispisao, na ekranu bi se pojavio crni kvadar, jer je trenutni volumen sastavljen od crno bijelih Dicom slika na čijim rubovima prevladava crna boja. Da bi se dobio zadovoljevajući prikaz, treba odrediti rubne konture usporedbom piksela sa najjačim i najslabijim intenzitetom.

Nakon toga, rubne točke spoje se izopovršinom boje kože, koja se naknadno još zagladi. Na kraju se dodaju omjer prikaza podataka, svijetlosni efekti i sl. treba spomenuti da je u programski kod ugrađen i filter koji sve piksele slabijeg intenziteta pretvara u potpuno crne, kako bi s izbjegao ranije spomenuti šum. Ukoliko snimke nebi bile filtrirane realni trodimenzionalni prikaz mogao bi se pojaviti nejasan, kao što je prikazano na slici 39.

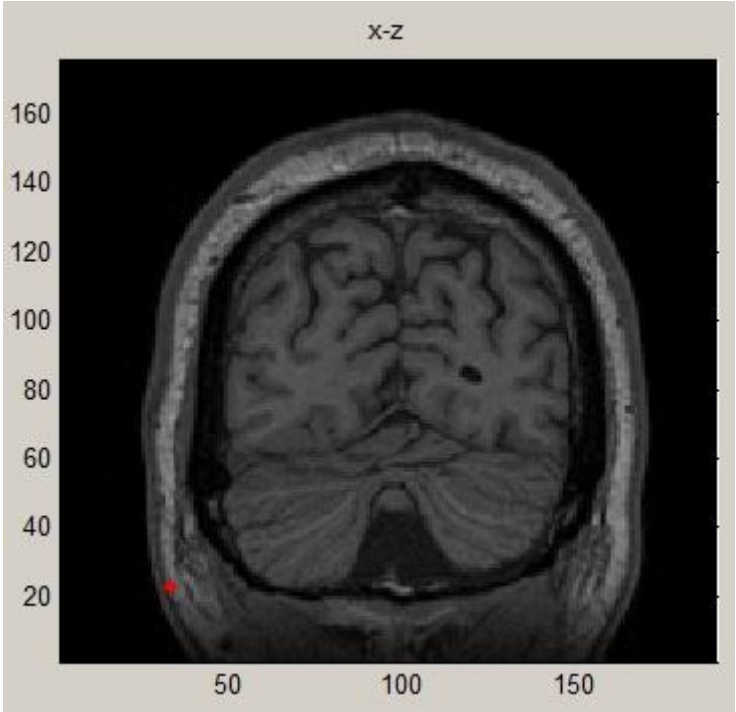


Slika 39. Utjecaj šuma na realni prikaz

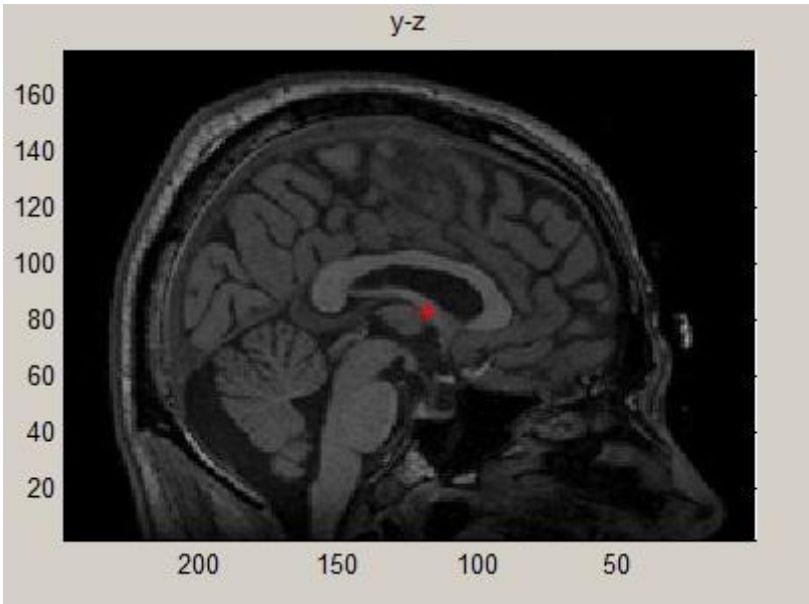
Ovakva pojava šuma nastaje kada se rubnom kontrom zahvate pikseli koji odstupaju od osnovnog niza podataka. U ovom slučaju oko lica. Ti pikesli bili su određenog intenziteta te je program mislio da je i to neki rub. Tako su pikesli zahvaćeni i izopovršinom, te je rezultat vrlo nejasan prikaz ljudske glave.

7.7. Prikaz prodora pravca putanje

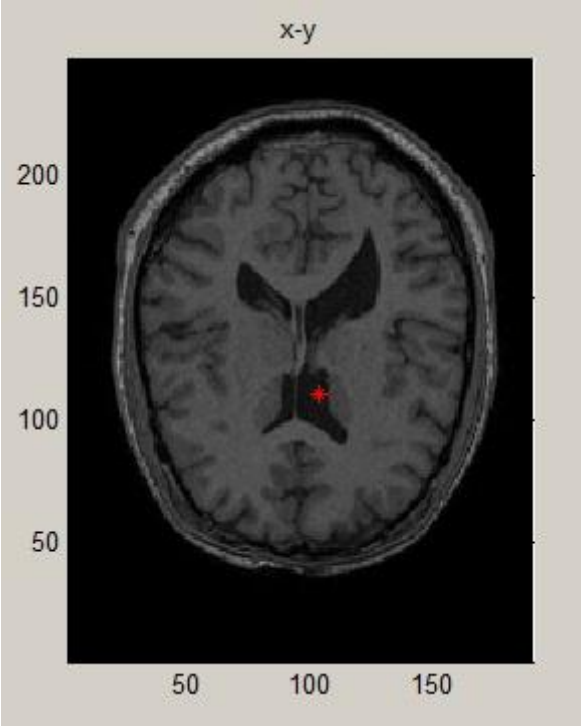
Program „Brainiac“ bio bi znanstveno prilično nekoristan, ukoliko nebi imao mogućnost prikaza točaka prodora medicinskog alata kroz pojedini presjek. U ovom slučaju prikaz prodora pravca koji se ranije generirao i koji predstavlja putanju medicinskog alata. Na taj način, u postupku planiranja same operacije, kirurg zadužen za operaciju, može detaljno proučiti na kojim mjestima kroz tkivo će proći njegov alat, igla ili što već. Ukoliko kirurg prepozna prodor alata kroz područje tkiva koje bi mogao trajno oštetiti, može započeti planirati drukčiju putanju uzimajući neku drugu točku, bilo ulaznu ili krajnju. Nakon novo odabrane putanje može ponovno prikazati točke prodora. Promjeni li se presjek na klizaču, program će se prilagoditi i pokazati točku prodora na tom presjeku. Točke prodora mogu se vidjeti na sva tri presjeka, a aktiviraju se pritiskom tipke sa znakom „X“ ispod svakog od presjeka. Primjer prikaza prodora na presjecima prikazan je na slijedećim slikama.



Slika 40. Prikaz prodora pravca na koronarnom presjeku



Slika 41. Prikaz prodora pravca na sagitalnom presjeku



Slika 42. Prikaz prodora pravca na aksijalnom presjeku

Prikaz prodora ispisuje se za svaki presjek posebno pritiskom na tipku „X“. Programski kod koji mogućuje ispis crvenih zvijezdica na mjestu prodora pravca (koji predstavlja putanju medicinskog alata) slijedi u nastavku.

Biti će prikazan kod za jednu os koji je alanlogan postupcima za ostale tri osi. Razlika je u ispisu koordinata koje se razlikuju od presjeka do presjeka.

```
axes(handles.axes1); %odabire aktivnu os
hold on; %zadržava prikazane podatke tako da se točka može ispisati preko
presjeka
x1=str2double(get(handles.edit7,'String')); %uzima vrijednost koordinata
y1=str2double(get(handles.edit8,'String')); %uzima vrijednost koordinata
z1=str2double(get(handles.edit9,'String')); %uzima vrijednost koordinata
x2=str2double(get(handles.edit13,'String')); %uzima vrijednost koordinata
y2=str2double(get(handles.edit14,'String')); %uzima vrijednost koordinata
z2=str2double(get(handles.edit15,'String')); %uzima vrijednost koordinata
%slijedi if petlja koja se sastoji od 8 istih dijelova. Jedina razlika je u
tome kako su točke odabrane. Primjerice ukoliko je prva točka viša po osi z
od prve slijedi jedna računica, a ukoliko je niža slijedi druga računica.
If petlja provjrava u kakvom su odnosu točke 1 i 2 te odabire pravi
matematički zapis.
if (z1<z2)&&(x1<x2)&&(y1<y2) %slučaj 1.
    razlika=abs(y2-y1); %računa broj presjeka između dvije odabrane točke
    s1 = evalin('base','s1'); %uzima trenutnu vrijednost klizača
    zslajsa = s1; %preimenuje trenutnu vrijednost klizača
    vaznoslajsa=abs(zslajsa-y1); %određuje referentni položaj prvog
presjeka u odnosu na prvu točku, tj. postavlja nulu koordinatnog sustava u
ishodište prve točke.
    z=z1+vaznoslajsa*(abs(z2-z1)/razlika); %matematička formula koja dijeli
udaljenost dvije točke sa brojem presjeka koji se nalaze između točaka te
množenjem sa položajem presjeka dobiva koordinate točke
    x=x1+vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*'); %crta točku u 3D prostoru
    assignin('base','xproba',z); %zapisuje koordinate točaka u bazu
podataka
    assignin('base','yproba',x); %zapisuje koordinate točaka u bazu
podataka

elseif (z1<z2)&&(x1<x2)&&(y1>y2) %slučaj 2.
    razlika=abs(y1-y2);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y2);
    z=z2-vaznoslajsa*(abs(z2-z1)/razlika);
    x=x2-vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);

elseif (z1<z2)&&(x1>x2)&&(y1<y2) %slučaj 3.
    razlika=abs(y2-y1);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y1);
    z=z1+vaznoslajsa*(abs(z2-z1)/razlika);
    x=x1-vaznoslajsa*(abs(x2-x1)/razlika);
```

```
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);

elseif (z1<z2)&&(x1>x2)&&(y1>y2) %slučaj 4.
    razlika=abs(y1-y2);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y2);
    z=z2-vaznoslajsa*(abs(z2-z1)/razlika);
    x=x2+vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);

elseif (z1>z2)&&(x1<x2)&&(y1<y2) %slučaj 5.
    razlika=abs(y2-y1);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y1);
    z=z1-vaznoslajsa*(abs(z2-z1)/razlika);
    x=x1+vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);

elseif (z1>z2)&&(x1<x2)&&(y1>y2) %slučaj 6.
    razlika=abs(y1-y2);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y2);
    z=z2+vaznoslajsa*(abs(z2-z1)/razlika);
    x=x2-vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);

elseif (z1>z2)&&(x1>x2)&&(y1<y2) %slučaj 7.
    razlika=abs(y2-y1);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y1);
    z=z1-vaznoslajsa*(abs(z2-z1)/razlika);
    x=x1-vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
```

```
    assignin('base','xproba',z);
    assignin('base','yproba',x);

else                        %slučaj 8.
    razlika=abs(y1-y2);
    s1 = evalin('base','s1');
    zslajsa = s1;
    vaznoslajsa=abs(zslajsa-y2);
    z=z2+vaznoslajsa*(abs(z2-z1)/razlika);
    x=x2+vaznoslajsa*(abs(x2-x1)/razlika);
    x=reshape(x,size(x,1)*size(x,2),1);
y=reshape(y,size(x,1)*size(x,2),1);
z=reshape(z,size(x,1)*size(x,2),1);
    plot3(x,y,z,'r*');
    assignin('base','xproba',z);
    assignin('base','yproba',x);
end
```

Srž prikazanog programskog koda je jednostavna matematička funkcija koja na osnovu podataka o dvije točke u trokooridnatnom sustavu, te položaju pojedinog presjeka, računa točku prodora putanje pravca na trenutnom presjeku. Prvi dio koda služi za postavljanje aktivnog koordinatnog sustava te zauzimanje podataka iz glavne baze Matlaba, kao što su položaj točaka i trenutni položaj presjeka u trodimenzionalnom koordinatnom sustavu.

7.8. Automatsko obnavljanje podataka

U pojedinim slučajevima analize podataka korisno je imati mogućnost automatskog obnavljanja podataka. Podaci koji se obnavljaju mogu biti, položaj presjeka u trodimenzionalnom prostoru, prodor pravca kroz presjeke, te prikaz točaka prodora na pojedinom presjeku. Program „Brainiac“ opremljen je takvim mogućnostima. Kako bi se automatsko obnavljanje podataka aktiviralo potrebno je označiti „auto“ opciju koja se nalazi u donjem lijevom dijelu programa. Programski kod za uspješno izvršenje automatskog obnavljanja slijedi u nastavku.

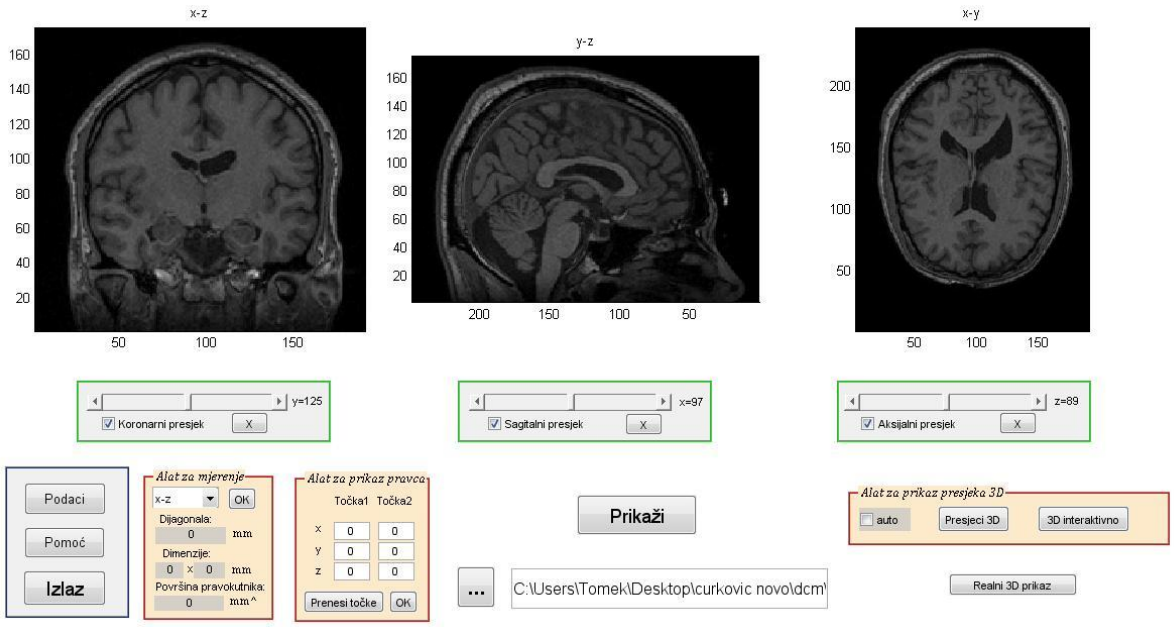
```
function varargout = checkbox10_Callback(~, ~, handles, varargin)
myplot(handles,[1 2 3]); %provjerava da li je označena opcija „auto“, i
aktivira funkciju koja ispisuje presjeke
```

```
if get(handles.checkbox10,'value'), %provjerava da li je označena opcija
„auto", te ako je aktivira tipke: „Presjeci 3D","OK"(za prikaz pravca)
    pushbutton3_Callback([],[],handles,[]);
    pushbutton9_Callback([],[], handles);
end


pause(1);
if get(handles.checkbox10,'value'), %provjerava da li je označena opcija
„auto", te ako je aktivira tipke za prikaz prodora
    pushbutton16_Callback([],[], handles);
    pushbutton17_Callback([],[], handles);
    pushbutton15_Callback([],[], handles);
end
```

Programski kod najprije naredi aktiviranje funkcije koja prikazuje presjeke u trokoordinatnom sustavu, tako prikazjući trenutne položaje presjeka, a zatim naredi tipkama, koje služe za prikaz putanje pravca i prikaz točaka presjeka, da se aktiviraju. Na taj način svaka radnja pomicanja presjeka biti će popraćena automatskim obnavljanjem njegovog položaja u trodimenzionalnom sustavu te će biti prikazana nova točka prodora ukoliko se razlikuje od prijašnje.

7.9. Aktivno sučelje programa „Brainiac"

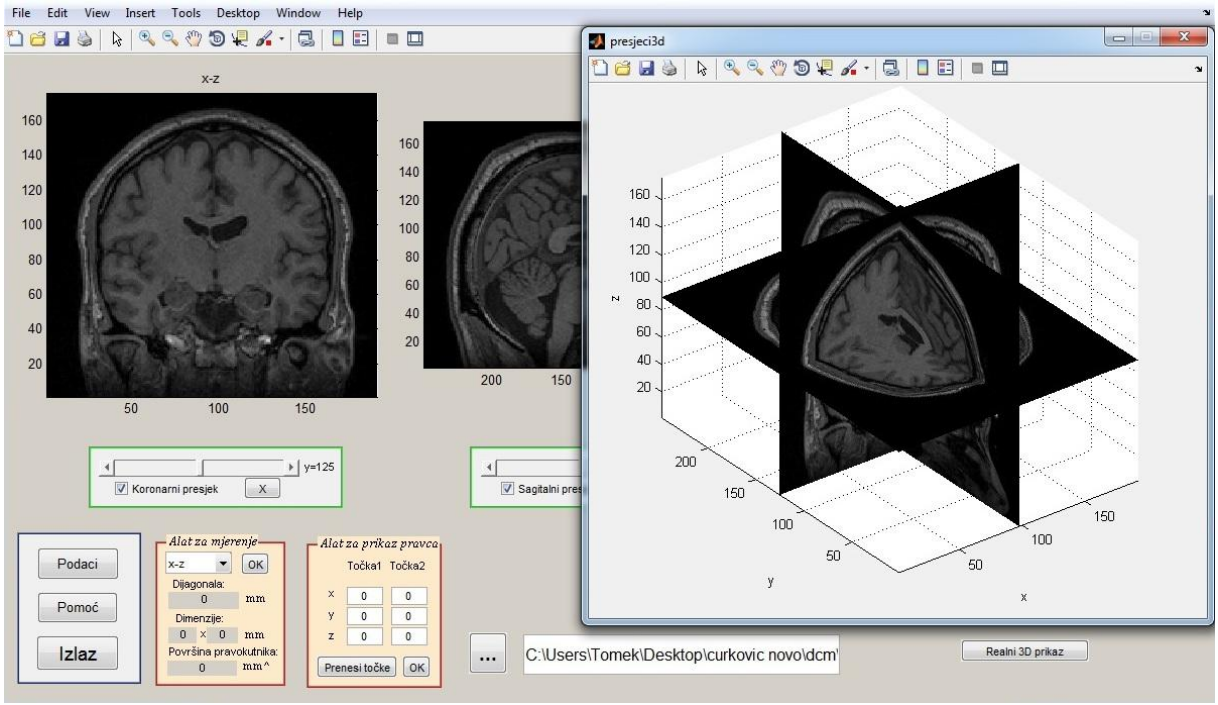
Do sada pokazane su pojedine funkcije programa, svaka zasebno. Vrijeme je da se pokaže program u funkciji prilikom rada DICOM snimkama. Snimke koje će se koristiti u slijedećem primjeru snimljene su u Kliničkoj bolnici Dubrava. Set se sastoji od 176 horizontalnih presjeka ljudske glave. Nakon što su slike na tvrdom disku računala sa kojeg se radi, nakon pokretanja programa potrebno je u traku za unos putanje snimaka upisati, ili pritiskm na tipku „..." naći set snimaka. Odaberu se sagitalni, koronarni i aksijalni prikaz te se tipkom „Prikaži" aktivira funkcija koja prikazuju presjeke. Nakon obavljenih akcija sučelje programa je kao na slici 43.



Slika 43. Sučelje programa netom nakon učitavanja snimaka

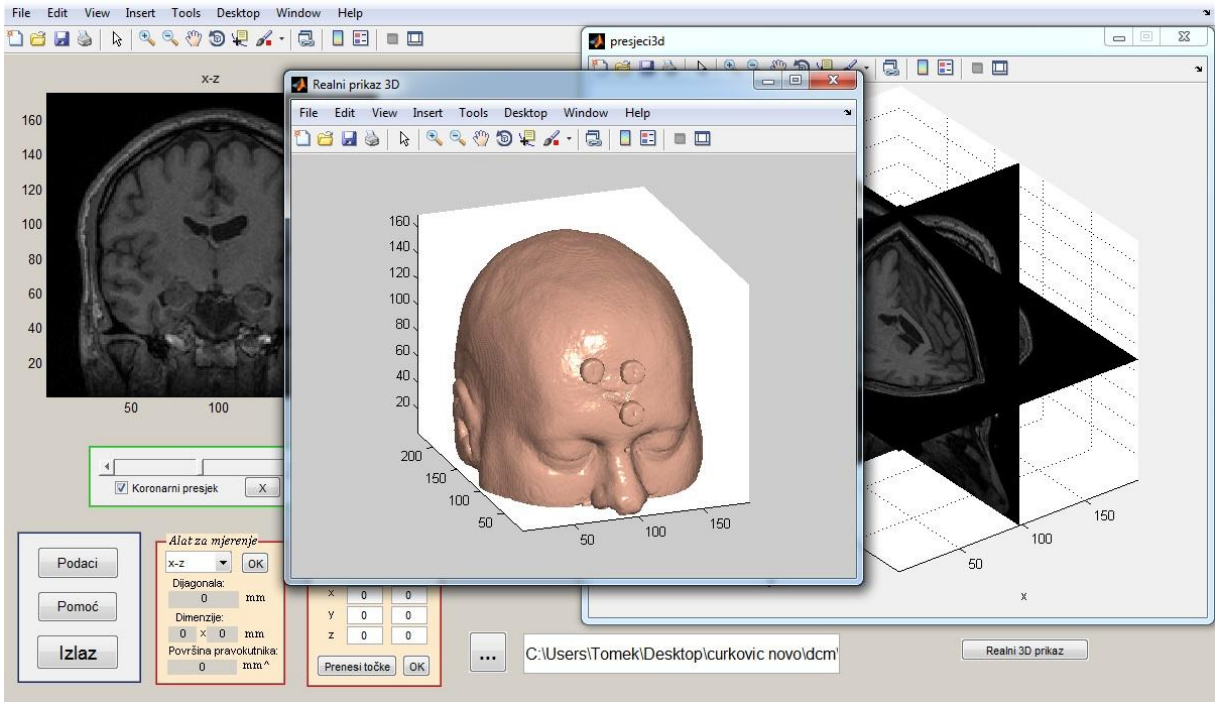
Daljnji koraci korištenja programa mogu biti različiti. Alatom za mjerenje može se mjeriti željena površina presjeka. Alatom za prikaz pravca odrediti putanja medicinskog alata, te prodori alata na pojedinim presjecima. Mogu se presjeci prikazati svi u jednom koordinatnom sustavu ili se može izraditi realni trodimenzionalni prikaz pacijentove glave. Naravno, postoje i tipke za informacije i pomoć, te izlaz.

Za potrebe demonstracije prikazati ćemo koronarni, sagitalni i aksijalni prikaz u jednom trokoordinatnom sustavu. Pritiskom na tipku „Presjeci 3D“ otvara se dodatni prozor kao na slici 44.



Slika 44. Sučelje programa nakon otvaranja dodatnog prozora sa presjecima

Pritiskom na tipku „Realni 3D prikaz“ otvara se još jedan dodatni prozor sa prilično realnom slikom pacijentove glave kako ona izgleda u stvarnosti. Prikaz je koristan prilikom odabira ulazne točke medicinskog alata.



Slika 45. Sučelje nakon otvaranja dodatnog prozora realnog 3D prikaza

Nakon što su se prikazali prozori svih presjeka u, jednom, trokoordinatnom sustavu, te prozor realnog 3D prikaza, stvoreni su svi uvijeti za proučavanje podataka, i planiranje medicinskog zahvata. Od ovog koraka pa na dalje možemo odabirati ulaznu i/ili krajnju točku prodora alata, izmjeriti udaljenost dvije željene točke na nekom od presjeka, ili izračunati odabranu površinu na presjeku. Ukoliko prikazani presjeci nisu na području interesa, pomakom klizača namješta se vrijednost presjeka u pojedinom smjeru. Nakon odabrane putanje i ispisa pravca pritiskom na tipke „X“, ispod svake od osi, pokrećemo funkciju koja računa točke prodora na presjecima. Ukoliko želimo da se korisničkom akcijom podaci obnavljaju automatski, potrebno je označiti funkciju „auto“ u donjem desnom kutu programa. Dodatni prozori mogu se pomicati te nebrojeno puta otvarati i zatvarati, tako dajući korisniku maksimalnu slobodu pregleda i upravljanja.

ZAKLJUČAK

Program „Brainiac“ moderna je programska podrška koja služi za planiranje operacija na moždanom tkivu u neurokirurgiji. Unosom snimaka sa magnetske rezonance, omogućava pregled presjeka u sve tri ravnine, planiranje putanje alata, te pregled prodora alata kroz presjeke. Za izradu programa nisu utrošena nikakava novčana sredstva, tek nešto vremena i volje.

Pregledom funkcija programa „Brainiac“ da se zaključiti da je to jedan suvremeni softverski paket kakav bi trebao imati važnu ulogu u unapređenju operacija na području neurokirurgije. Danas već postoje mnogi uređaju i programske podrške bez kojih bi neki medicinski zahvati bili nezamislivi. Neosporno je da čovjek i tehnologija, u obliku robota i uređaja, moraju surađivati kako bi izvodili složene operacije koje mogu biti razlika između života i smrti.

Znanstveno područje robotike te robotika u medicini vrlo je perspektivna grana znanosti koja će, vjerujem, u budućnosti igrati važnu ulogu u produljenju životnog vijeka čovjeka, te sve većem broju uspješno provedenih, vrlo složenih, operacija. Program „Brainiac“ služi kao primjer da je za razvoj vrhunskih programskih podrška primjenjivih u medicini za planiranje vrlo složenih i opasnih medicinskih zahvata, neophodno ulagati u obrazovanje i razvoj stručnjaka na području robotike. Tko zna, možda jednoga dana svjedočimo prvoj uspješnoj operaciji, koju je robot sam proučio, planirao i obavio. Takav način obavljanja operacija bio bi revolucionaran. Danas kada je, praktički, svo znanje svijeta nadohvat ruke, razvojem interneta i komunikacijskih tehnologija, program „Brainiac“ je dokaz da nas od fikcije do stvarnosti dijeli tek nekoliko znanstvenih „koraka“.

LITERATURA:

[1]. http://en.wikipedia.org/wiki/DICOM, 22.11.2012.

[2]. http://www.bayshorewomenshealth.com/obgyn-services/da-vinci-robotic-surgery/, 22.11.2012.

[3]. http://hr.wikipedia.org/wiki/Robotska_kirurgija, 22.11.2012.

[4]. http://blogs.computerworld.com/cloud-computing/, 23.11.2012

[5]. http://titan.fsb.hr/~mvrdolja/matlab/node3.html, 23.11.2012.

[6]. Z. Valter, K. Miklošević, Ž. Špoljarić; 2008/2009. Uvod u MATLAB.